

AUTO FOLLOWER CART

By

AMRO BATWA	1741347	COE
AHMED PATWA	1845044	COE
ALFAHD FELEMBAN	1742444	COE

TEAM NO.: 06 FALL-2021 INTAKE

Project Advisor

PROF. MOHAMMED BILAL

Project Co-advisor

PROF. KHALID MUNAWAR

Project Customer

PROF. MOHAMMED BILAL

**DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
FACULTY OF ENGINEERING
KING ABDULAZIZ UNIVERSITY
JEDDAH – SAUDI ARABIA**

JUN 2022 G – SHAWWAL 1443 H

AUTO FOLLOWER CART

By

AMRO BATWA	1741347	COE
AHMED PATWA	1845044	COE
ALFAHD FELEMBAN	1742444	COE

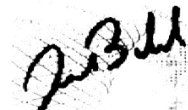
TEAM NO.: 06

FALL-2021 INTAKE

**A senior project report submitted in partial fulfillment of the
requirements for the degree of**

**BACHELOR OF SCIENCE
IN
ELECTRICAL AND COMPUTER ENGINEERING**

CHECKED AND APPROVED (ADVISOR):



SDP EVALUATOR: _____

**DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
FACULTY OF ENGINEERING
KING ABDULAZIZ UNIVERSITY
JEDDAH – SAUDI ARABIA**

JUN 2022 G – SHAWWAL 1443 H

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ
وَالصَّلَاةَ وَالسَّلَامَ عَلَى خَيْرِ الْوَرَى عَدُّ
الْحَصَى وَالرَّمْلِ وَالتُّرَى
وَعَلَى آلِهِ وَأَصْحَابِهِ وَمَنْ تَبَعَ هَدَاهُمْ
وَاهْتَدَى

نَحْمَدُ اللَّهَ الْعَلِيِّ الْقَدِيرَ الَّذِي قَدَّرْنَا عَلَى
إِنْهَاءِ هَذَا الْمَشْرُوعِ فِي الْوَقْتِ الْمَحْدَدِ

**To our parents and families, and our teachers who facilitated the world to
us ,,,**

ABSTRACT

Auto Follower-Cart

In many situations, one finds himself in need to transport a group of objects from one point to another inside closed environments. As an example, you might need to transport a water tank from the entrance of the apartment to the kitchen, or from the elevator to your office at work. However, having to carry, pull, or push heavy objects between two points inside closed environments, consumes time, effort, and can affect long term health.

The customer wants are a robot capable of carrying at least 50kg, and follows the user or can be controlled in such a way to eliminate the effort and health risks. Such a project aims to enhance the overall long-term health of individuals, by eliminating their need to carry, pull, or push heavy objects inside closed environments. Three alternative designs were suggested, and the final design was to implement the robot on a base of electric hoverboards. The robot locates the user using a camera that detects Aruco marker (a simple marker that is similar in shape to QR-Code) that the user is carrying on his back (following mode), or in his hands (pushing mode). The design also features a very simplified phone app for positioning the robot in the suitable place for loading and unloading.

The algorithm used for following uses adaptive speed depending on the distance between the robot and the user, increasing the speed when the robot gets far away. It also keeps safe distance from the user to insure safety, and can detect obstacles and stop at the sight of any. Using Aruco markers for detecting enabled for robust, smooth, and accurate detection, with less than 1cm error in distance estimation. The hoverboard used also proved to be able to provide consistent speeds under different loads. The use of Aruco also enabled the project to run on small devices, like the Raspberry Pi. Also, the pushing mode presented by the team presents a convenient way to control the robot, without having to exert any effort in actual pushing. Also, 3D printed components are used for aesthetics.

This project is expected to have positive impacts on society, locally and globally. As it can change the habits related to transporting heavy objects. It is also expected to introduce the concept of robotics to the general consumers and make it easy to accept the integration of robotics in everyday activities.

Index Terms — AutoCart, Robot, loads, objects, indoor environments, hoverboards, camera, Aruco marker, pose estimation, 3D printing.

ACKNOWLEDGEMENT

Our great thanks to **Prof. Mohammed Bilal** for his support and for generously sharing his knowledge with us, and for providing us a lab to work at in the university.

Our thanks to **Eng. AbdulRauf Norwali** and **Eng. Talha Fattani** from the mechanical engineering department at KAU for their help on the mechanical modifications of the project.

TABLE OF CONTENT

.....	III
ABSTRACT	V
ACKNOWLEDGEMENT	VI
TABLE OF CONTENT	VII
LIST OF FIGURES	IX
LIST OF TABLES	X
CHAPTER – 1 INTRODUCTION	1
1.1 ABOUT THE PROJECT	1
1.2 BACKGROUND.....	1
CHAPTER – 2 CONCEPTUAL DESIGN.....	2
2.1 SITUATION DESCRIPTION.....	2
2.2 DEFINING THE PROBLEM.....	2
2.3 PROJECT OBJECTIVES.....	3
2.4 APPLICABLE ENGINEERING STANDARDS	3
2.5 CONSTRAINTS.....	4
2.6 PRODUCT DESIGN SPECIFICATIONS (PDS).....	4
2.7 LITERATURE REVIEW.....	6
2.8 ANALYZING ALTERNATIVE SOLUTIONS	8
2.8.1 1st Alternative.....	9
2.8.2 2nd Alternative.....	11
2.8.3 3rd Alternative.....	13
2.8.4 Alternatives Evaluation.....	14
2.9 MATURING BASELINE DESIGN	15
CHAPTER – 3 PRODUCT BASELINE DESIGN.....	16
3.1 BLOCK DIAGRAM	16
3.2 SYSTEM DESCRITPION.....	17
3.2.1 Circuit component specifications	17
3.2.2 Flowcharts for software blocks.....	18
3.2.3 Mechanical specifications of the case.....	19
3.2.4 Possible aesthetics	20
3.2.5 Operating Instructions.....	22
CHAPTER – 4 IMPLEMENTATION	24
4.1 MECHANICAL MODIFICATIONS	24
4.2 COMMUNICATION AND HOVERBOARD CONTROL	26
4.2.1 First trial.....	26
4.2.2 Second trial.....	27
4.3 CAMERA SETUP.....	29
4.3.1 First trail (Web cam).....	29
4.3.2 Second trail (Bi-cam).....	29
4.3.1 Third trail (OAK-D).....	30
4.4 USER DETECTION USING ARUCO MARKER.....	30
4.4.1 First trial	30
4.4.2 Second trial.....	32
4.4.3 Third trial.....	32
4.5 FOLLOWING ALGORITHMS	33
4.5.1 First trial	33
4.5.2 Second trial.....	34
4.5.3 Third trial.....	37
4.6 THE TRANSITION TO SINGLE-BOARD COMPUTER.....	37
4.6.1 First trial: Raspberry Pi 3.....	38
4.6.2 Second trial: solving undervoltage problem.....	38
4.6.3 Third trial: transitioning to Raspberry Pi 4.....	38

4.7	USER INTERFACE	39
4.7.1	First trial	39
4.7.2	Second trial.....	39
4.8	3D PRINTING	40
4.9	FINAL PRODUCT	41
CHAPTER – 5	RESULTS, DISCUSSION, AND CONCLUSIONS	44
5.1	RESULTS AND DISCUSSION.....	44
5.1.1	Aruco detection accuracy results	44
5.1.2	Hoverboard speed results under different weights.....	47
5.1.3	Communication results	48
5.1.4	Results of testing after final assembly.....	50
5.2	EVALUATION OF SOLUTIONS	51
5.2.1	Technical Aspects.....	51
5.2.2	Environmental Impacts.....	52
5.2.3	Safety Aspects.....	52
5.2.4	Financial Aspects	53
5.2.5	Social Impacts.....	54
5.3	CONCLUSIONS	54
REFERENCES		56
APPENDIX – A: VALIDATION PROCEDURES		59
EXPERIMENT 1		59
Introduction.....		59
Objectives.....		59
Variables.....		59
Constants.....		60
Assumptions.....		60
Safety.....		60
Experiment tools.....		60
Obtaining distance work plan.....		61
Obtaining angle work plane.....		61
Collected data		62
Data analysis.....		64
Discussion and conclusion.....		65
Considerations of Engineering Standards		66
Experiment References		67
EXPERIMENT 2		68
Introduction.....		68
Tools.....		68
Setup and work plan		69
Data Collection.....		69
Conclusion.....		71
EXPERIMENT 3		72
Introduction.....		72
Objectives.....		73
Experimental Setup.....		74
Tools.....		74
Work Plan		75
Assumptions:.....		75
Issues and experimental hazards:		76
Collected Data.....		76
Conclusion:.....		78
EXPERIMENT 4		79
Introduction.....		79
Objectives.....		79
Tools.....		79
Setup and work plan		80
Assumptions:.....		80
Issues and experimental hazards:		81

Collected Data.....	81
Conclusion:.....	82
APPENDIX – B: SELF ASSESSMENT CHECKLIST	83

LIST OF FIGURES

Figure 1: Block Diagram - 1st Alternative	9
Figure 2: Block Diagram – 2nd Alternative	11
Figure 3: Block Diagram - 3rd Alternative.....	13
Figure 4: Modified block diagram of baseline design	16
Figure 5: Simplified software flowchart	19
Figure 6: The mechanical structure of the project as provided by the manufacturer ^[10]	20
Figure 7: Mechanical structure (raw before modifications)	21
Figure 8: From left to right: Raspberry Pi compartment, camera stand, and ESP compartment	21
Figure 9: The battery in its compartment, hidden under the mechanical structure	22
Figure 10: Comparing hoverboard width with mechanical structure width	24
Figure 11: An image after disassembling the two parts of the hoverboard showing the small rod to be replaced.....	25
Figure 12: An image of the structure after attaching the hoverboard	26
Figure 13: Hoverboard mainboard ^[11]	27
Figure 14: ESP-8266 Circuit.....	28
Figure 15: ESP-8266 After soldering	28
Figure 16: Webcam used at the start of the project	29
Figure 17: Stereo Camera and its output	29
Figure 18: OAK-D Camera.....	30
Figure 19: Example from the calibration process	31
Figure 20: The axis of Aruco marker.....	32
Figure 21: Explaining the field of view of the camera	35
Figure 22: Explaining steering algorithm	36
Figure 23: User Interface	40
Figure 24: First camera stand – The final camera stand	41
Figure 25: The final project - general view	42
Figure 26: Final project - Raspberry Pi	43
Figure 27: One of the team members showing the suit	43
Figure 28: Plot of the error in distance estimation.....	45
Figure 29: plot of "rotation angle" estimation error.....	47
Figure 30: Average speeds under different loads	48
Figure 31: Data collection of communication testing.....	49
Figure 32: Testing the product, stopping delay chart	50
Figure 33: Axis of Aruco marker.....	60
Figure 34: A graph to explain the setup for experiment 1 (distance)	61
Figure 35: An example of the screen output (Experiment 1, angle).....	62
Figure 36: Plot of the error amount in distance measurements	65
Figure 37: Plot of the error amount in angle measurements.....	66
Figure 38 weights used in 2 nd experiment	68
Figure 39 Second experiment, example of a load of 10 kg	68
Figure 40: Experiment 2, Average speed at different weights	70
Figure 41: Experiment 2, Error for different weights. x-axis weights, and y-axis speed ...	70
Figure 42: Third Experiment: Experiment Setup	74

Figure 43: Third Experiment, NetSpot Application and measuring wheel	75
Figure 44: Third Experiment, dBm readings chart	77
Figure 45: Third Experiment, dBm readings with signal strength levels	77
Figure 46: Fourth experiment, stopping delay chart	82

LIST OF TABLES

Table 1: Musts and wants	5
Table 2: Morphological Chart - Design Space Generation.....	8
Table 3: Three Alternative Designs	9
Table 4: Total cost - 1st Alternative	10
Table 5: Total cost - 2nd Alternative	11
Table 6: Total cost - 3rd Alternative.....	13
Table 7: Evaluation of three alternatives	14
Table 8: Main electric components specifications.....	17
Table 9: Results of distance estimation error.....	44
Table 10: Results of "rotation angle" estimation error	46
Table 11: Project Cost Analysts.....	53
Table 12: First Experiment, Distance Measurements - First batch.....	62
Table 13: First Experiment, Distance Measurements - Second batch	63
Table 14: First Experiment, Distance Measurements - Third batch	63
Table 15: First Experiment, Angle measurements.....	63
Table 16: First experiment, Error calculations for distance measurements.....	64
Table 17: First experiment, Error calculations for angle measurements	64
Table 18: Experiment 2, data collection	69
Table 19: Third Experiment, Signal Strength Quality	72
Table 20: Third Experiment, Test: open space using laptop	76
Table 21: Third Experiment, Test: Narrow corridors using phone.....	76
Table 22: Fourth experiment, data collection (10 meters distance).....	81
Table 23: Fourth experiment, data collection (20 meters distance).....	81

CHAPTER – 1 INTRODUCTION

1.1 ABOUT THE PROJECT

An issue that people might face in their everyday activities in indoor environments, is the necessity to transport (carry, pull, or push) heavy (or many) objects between two points. This, in time, might reflect negatively to their joints, and back. Such an issue can be eliminated or heavily reduced by using the AutoCart. and for that we consider the AutoCart a contribution to the long-term health of its users.

The Auto Follower-Cart or, AutoCart in short, is a semi-autonomous robot aimed for in-doors use. The AutoCart uses Computer Vision technology, which is a field of artificial intelligence enabling computers to extract information from images or videos, in our case, Computer Vision is an important part of our project, processing the camera's input will give the AutoCart the means to identify its user who it is supposed to keep up with. The AutoCart will also be designed to continuously sense for obstacles while following the user and stop at the sight of any obstacle.

1.2 BACKGROUND

As a team we looked for medical research about the effects of lifting heavy objects to the health and well-being of the human body. According to resources, frequent necessity to lift heavy objects that weight more than 25 kg can cause lower back pain, the study showed that those individuals who are exposed to lifting showed an increase of 4.32% of incidence of lower back pain. Since one of our main objectives of this project is to contribute the health and well-being of humans, either workers or in a home environment, these findings further motivate our team to put hard work and effort to ensure the effectiveness of the final product to these aspects.^[1]

CHAPTER – 2 CONCEPTUAL DESIGN

2.1 SITUATION DESCRIPTION

For the current situation, the need arises to transport items, in office environments in particular, and closed environments in general. Taking the university office environment as an example, the current system consists of items that needs to be transferred, and an individual responsible for transporting it between two points inside the building. The items could be single, multiple, heavy, and/or light objects. The responsible individual will have to exert force to transport these objects, whether by carrying, pushing, or pulling. As the number of items and their weight increases, it causes inconvenience for the responsible individual and makes it hard to carry it all at once. As the weight increases, it will lead to long-term health issues if repeated frequently.

2.2 DEFINING THE PROBLEM

Based on the situation described in the previous section, the initial problem statement is as follows:

Transporting objects from the entrance of the faculty to offices takes a lot of effort

This statement arises few questions. First, does it take effort only from the entrance to offices? Do all objects take effort to transfer? Taking these questions into account, consider that, the revised problem statement:

Transporting heavy objects between distant points in the faculty takes a lot of effort

This statement suggests that this problem is only inside the faculty building. How about other indoors environments? Also, is it only heavy objects that are troublesome? What is meant by effort, The final definition:

Repeatedly carrying, pushing, or pulling heavy objects between two points inside closed environments, for long durations, consumes time, effort, and affects long term health

2.3 PROJECT OBJECTIVES

The following objectives contribute to the transition from the present state "The inconvenience of transporting objects inside closed environments" to the desired state "efficiently transporting large number of items and/or heavy items inside closed environments".

For **higher-level** objectives, the project aims for the following points:

- Enhance the overall long-term health of individuals, and support individuals with health constraints, by eliminating their need to carry heavy objects inside closed environments.
- Enable individuals to focus on other tasks while transporting objects in closed environments, and thus increase their productivity.

For **lower-level objectives**, the project aims for the following points:

- To provide a robot that is capable of identifying the user and following him, and capable of being controlled remotely as well.
- To provide a product capable of carrying heavy objects of at least 50 kilograms.

On technical side, the technical objectives for the project are:

- Stop at a safe distance of 1 meter from the user.
- Simplified user interface, with simple power level display.
- Quick and accurate response to the control signals.

2.4 APPLICABLE ENGINEERING STANDARDS

Engineering standards applicable for this project are written in this section, in accordance with standards and regulations provided by Saudi Standards, Metrology and Quality Organization (SASO).^[2]

These applicable standards are as following:

- 1- The supplier shall determine the functioning of machinery and the safety components that include their intended use, and define any expected

misuse. This will be achieved by providing the user with a clear and concise manual.

- 2- Remotely controlled machinery shall be designed and installed in such manner that they respond only to signals sent from the relevant control units. This is achieved by designing the robot to only respond to its particular controller and not respond to multiple ones.
- 3- Pedestrian-controlled machinery control systems shall be designed in such manner that reduces the risks arising from unintended movement of the machine towards the driver. This is applied through designing the system to continuously detect the distance from the user and stop at safe distance.
- 4- The machinery shall be designed in such a way that the battery can be physically disconnected, which can be applied through placing the battery in an accessible, safe compartment.

2.5 CONSTRAINTS

In this section, we address the constraints of the project being designed. The design of the product must stay within these boundaries. These constraints are:

- Time constraint: the project must be delivered within the allowed period of SDP starting Fall 2021.
- Cost constraint: Total spending on research and development of the prototype must not exceed 6000 SAR.
- Environment constraint: The project is meant to be operated only inside closed environments of a single floor or have elevators.
- Operation constraint: The user of the robot must stay within close range to the robot (~1 to 2 meters) and within the line of sight of the robot.

2.6 PRODUCT DESIGN SPECIFICATIONS (PDS)

Based on information provided by the customer and information provided in previous sections, the specifications of the product design were determined and

the scope was defined. This product is to be a robot with a main function of being capable of carrying and transporting objects inside closed environments and move on its own semi-autonomously (with a help of an operator that it will follow). Before addressing the specifications, we show some assumptions made by the team, which are:

- While using the product (the robot), the user will be walking, or idle in his place, but not running or jogging.
- Human average speed is assumed not to exceed 2.5m/s.
- The assistance of mechanical engineering department is acquirable for the mechanical parts of the project.
- Parts needed in the project are available locally or can be delivered within time.
- The control interface will be simple enough for average users.

The following table summarizes the compulsory in-scope items (musts) and the out-of-scope items (wants) of the product.

Table 1: Musts and wants

Musts	Wants
The minimum weight support of the structure is 50kg	Weight support of the structure up to 100kg
Two modes of operation: following mode and remote-controlled mode	Loading and unloading mechanisms
Must keep up with the user's speed (not to exceed 2.5m/s)	Various accessories to facilitate carrying specific objects (i.e. gas containers, luggage, water tanks ... etc.).
Provide an indicator to know when to charge the battery	Battery lasts up to 8 hours per charge
Battery lasts for at least 2 hours of operation per charge	
Must keep a minimum safe distance of at least 1 meter from the user when following him	

2.7 LITERATURE REVIEW

To further understand the solution and how it could be implemented, the team searched for similar design problems. One of the results was a paper published by the IEEE, 2016, titled "Vision-Based Human Tracking Control of a Wheeled Inverted Pendulum Robot" discussed the design of vision-based adaptive control for a wheeled inverted pendulum robot. For mechatronic design, the system uses two differential driving wheels. The robot is powered with two 24V rechargeable lithium batteries in series and consists of two 48V dc servomotors with gearbox and two stamped steel wheels with 16-in tires. For the vision system, the tracking was achieved by combining OptiTrack Flex 3 camera with Microsoft Kinect camera, where the first is used for target capture and the second for distance detection. The algorithm for target detection on RGB image extracts the target coordinates from OptiTrack camera and transforms them into RGB image captured by the Kinect sensor.^[3]

The team also found a paper published on 2016, that focus entirely on human tracking algorithm, titled "A Computationally Low-Cost Vision Based Tracking Algorithm for Human Following Robot". In their work, the robotic platform is a two-wheel differential drive system with feedback system, and additional rear free moving wheel. Instead of detecting the human operator, their algorithm detects a unique visual tag using a camera, that help identify the user. The reason for detecting a visual tag instead of people is the difference in the computational cost. According to the paper, detecting humans and differentiation between them and the user is extremely costly and demanding on robots that have limited resources and require real-time tracking. The tag consists of four different colors in equal proportions and adjacent to each other. What makes the tag unique in its surrounding is the fact that there is very low probability of same color combination in the picture frame. In addition to the camera, ultrasonic modules are used to measure distance. The image processing algorithm ran on Raspberry Pi2 model B, and based on the experimentation results, the overall time of acquiring image, and processing it with sensors data, was 312.7ms.^[4]

Another paper published 2017, titled "A Person-following Robotic Cart Controlled via a Smartphone Application: Design and Evaluation". As the title implies, their

designed robot can be controlled via phone and in follower mode as well. The cart uses ultrasonic sensors to locate and follow the user. The design has the constraints of operating indoors only, where the ground is flat and clean. It provides 40kg weight support at walking speed, but it can carry up to 80kg which reduces its speed. It is constructed from steel tubing with cargo area of 60by80 cm. The cart has two coaster wheels in the front with two DC motors. The sensing system is based on 6 ultrasonic sensors, pointing in different angles and mounted in different locations, to be able to detect the presence of ground and detect the user and keep constant distance from them. An Arduino Mega is used to control the sensors and acquire data from it and control the motors. The paper show testing results regarding motor drive and electrical components but no results regarding user tracking system.^[5]

Another project titled " Person Following Robot Using Selected Online Ada-Boosting with Stereo Camera" which was published on 2017, introduces a robust way to detect humans for follower robots. It addresses the problems of partial covering of the target, collusion, pose-changes. Instead of using complex tracking algorithms to improve robustness and sacrificing real-time operation, they achieved this via updating the AI model in the real time to better suit the current situation and environment. The updating happens by giving the model positive and negatives example in Real time.^[6]

A different approach is suggested in a paper entitled " Development of human tracking technique for mobile robot using QR code and 2D lidar sensor" published in 2020, addressed that using deep neural network achieve high recognition accuracy in various environments needs a lot of processing. This can be disadvantage and requires the use of high-performance computers. To overcome this issue, the paper suggests the use of 2D sensors such as lidar, in addition to a camera to detect QR code that the user carries. However, the paper discusses a disadvantage of this method, which is that QR code could be obscured by other people, and recognition rate of QR may be affected if the camera is shaken.^[7]

2.8 ANALYZING ALTERNATIVE SOLUTIONS

At this stage, the team tried to find different alternative designs for the problem at hand. Having multiple alternative designs helps the team not getting stuck at non-optimal solution. A morphological chart is employed to generate design space. This is shown in Table 2.

Table 2: Morphological Chart - Design Space Generation

Function	Mean		
Move and follow	Wheels	Caterpillar tracks	Mechanical legs
User authorization	QR code	Predefined Aruco marker	Face recognition
Detect user position indoors	Camera detecting and recognizing QR code	Camera detecting and recognizing Aruco marker	Face and body detection
User interface	Dedicated remote-control device	Voice commands	Phone app
Processing and decision making	Single board computer	Microcontrollers	Phone

The morphological chart introduces the functions that the project should do, and the means by which these functions can be achieved. The table shows 5 major functions expected by the solution. The morphological chart produces a big number of combinations. This number needs to be reduced, and thus different combinations have to be eliminated. The team continued with the process of narrowing the design space and three alternative designs were chosen as following:

Table 3: Three Alternative Designs

	Alternative 1	Alternative 2	Alternative 3
Movement	Wheels	Wheels	Wheels
User authorization	Predefined Aruco marker	QR code	Face recognition
Detect user position indoors	Camera detecting and recognizing Aruco marker	Camera detecting and recognizing QR code	Face and body detection
User interface	Phone app	Phone app	Phone app
Processing and decision making	Single board computer	Microcontrollers	Single board computer

The three alternatives shown in Table 3 will be discussed in the next subsections.

2.8.1 1st Alternative

Use Wheels for movement. A camera will detect an Aruco marker to follow the user, and it will authenticate its user by detecting the ID of that Aruco marker. A phone app is used for remote control. All the processing is done using single board computer.

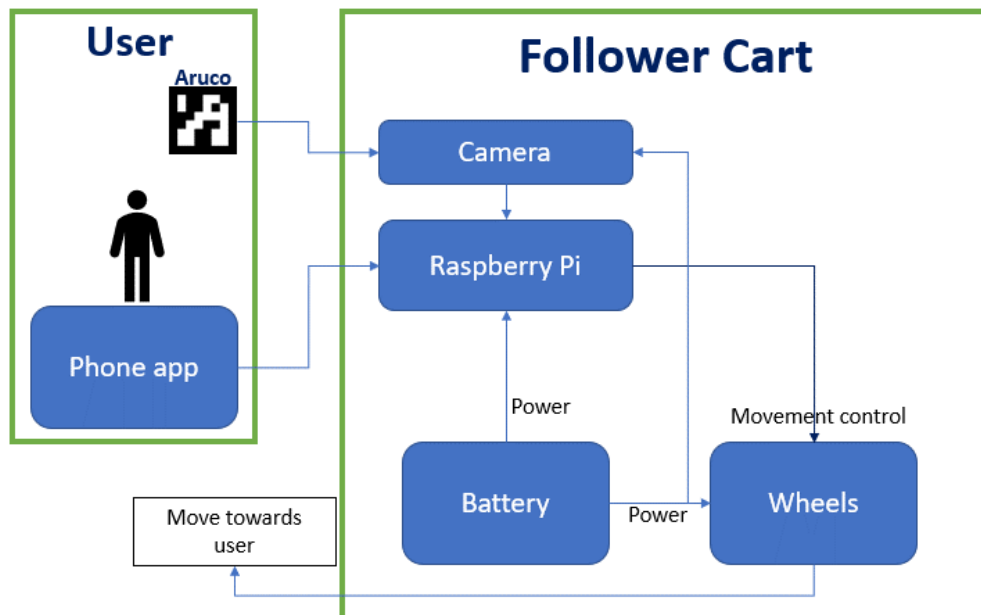


Figure 1: Block Diagram - 1st Alternative

Table 4 shows the total estimated cost of developing this alternative roughly based on components price.

Table 4: Total cost - 1st Alternative

Component	Estimate price (SAR)
Electric hoverboard (for the wheels and motors)	~ 600
Mechanical structure	~ 550
Camera	~ 300
Custom built holder for Aruco marker	~ 25
ESP module for wireless communication	~ 30
Raspberry Pi module	~ 200
Battery	~ 170
Total	~ 1875

Pros:

- Aruco markers are robust and provide user orientation and pose estimation
- Single board computers enable to use higher level language libraries
- Phone app is simple to use for most users

Cons:

- Require flat surfaces to operate
- If the Aruco marker is partially obstructed, the camera will lose track of the operator
- Single board computers cost more than simple microcontrollers

This alternative is balanced between simplicity to the user and availability of parts and easier maintenance. Justified using wheels which are easier and cheaper to repair and more cost efficient, and the use of Aruco marker solves user's orientation change.

2.8.2 2nd Alternative

Use wheels for movement. A camera will detect and recognizing QR code for user following. The information encoded in the QR code are used for user authentication. A phone app is used for remote controlling the robot. All the processing is done using a microcontroller.

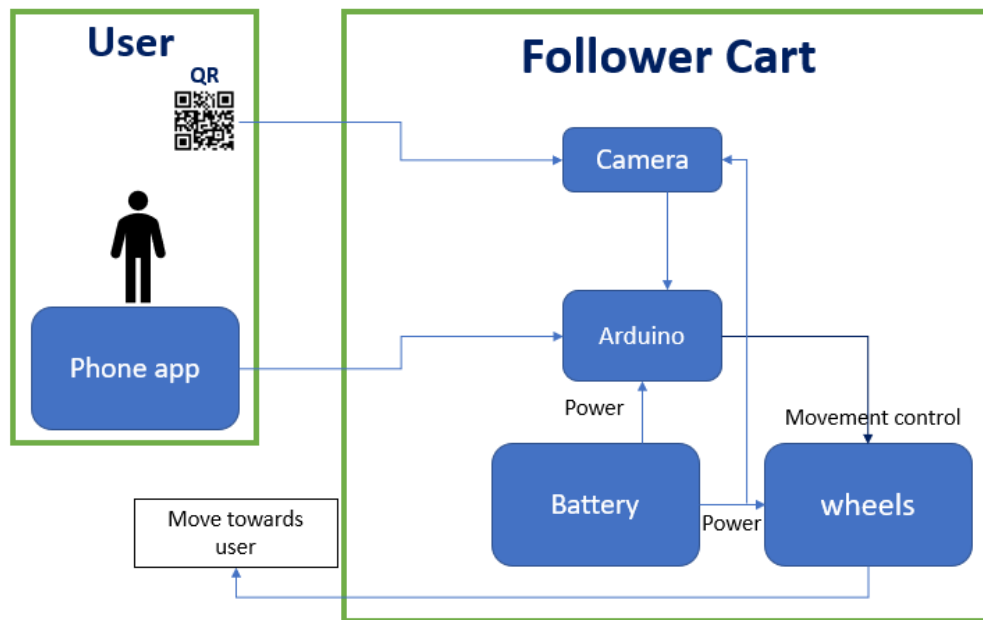


Figure 2: Block Diagram – 2nd Alternative

Table 5 shows the total estimated cost of developing this alternative roughly based on components price.

Table 5: Total cost - 2nd Alternative

Component	Estimate price (SAR)
Electric hoverboard (for the wheels and motors)	~ 600
Camera	~ 300
Mechanical structure	~ 550
Simple designed tag (to hold QR code and attach to user's pocket)	~ 25
ESP module for wireless communication	~ 30
Arduino Uno	~ 135
Battery	~ 170
Total	~ 1810

Pros:

- QR code are quick to be scanned even on microcontrollers.
- The encoded information inside the QR can help to provide more customizability.
- Microcontrollers cost less in general.

Cons:

- Even though microcontrollers are more suitable for end products, they are still limited in power.
- Unlike Aruco markers, there are no quick solutions for pose estimation using QR code. It will be harder to estimate distance and rotation through it.

2.8.3 3rd Alternative

Use wheels for movement. Use face recognition for authentication, and camera that detects the body of the user to follow him. The user can use a mobile app for remote control. The whole processing will be carried through a single board computer.

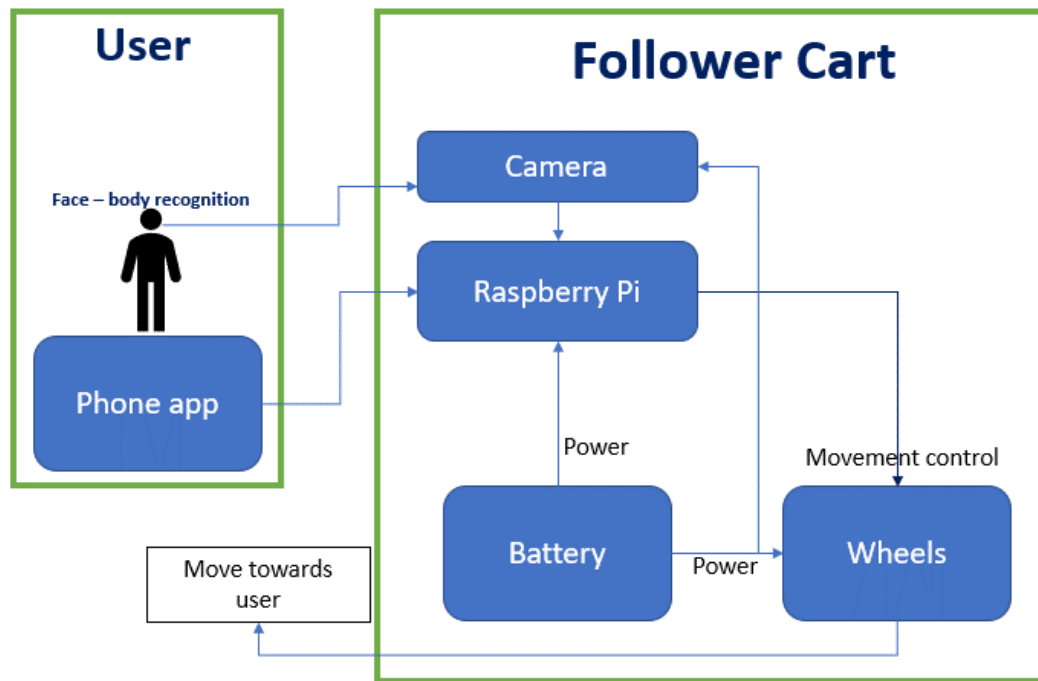


Figure 3: Block Diagram - 3rd Alternative

Table 6 shows the total estimated cost of developing this alternative roughly based on components price.

Table 6: Total cost - 3rd Alternative

Component	Estimate price (SAR)
Electric hoverboard	~ 600
Mechanical structure	~ 550
High-level Camera	~ 600
ESP-modules for wireless communication	~ 30
Raspberry Pi 3	~ 200
Battery	~ 170
Total	~ 2150

Pros:

- Excellent detection of user orientation through body recognition.
- User won't get out of sight if partially blocked by an obstacle

Cons:

- High computation load on the processing unit due to body tracking.
- Requires developing sophisticated algorithms
- The camera might mix multiple people in the frame and follow someone else.

2.8.4 Alternatives Evaluation

All the alternatives satisfy the in-scope items. So to choose one final alternative, evaluation of these alternatives is carried out by checking the pros/cons for each alternative, in addition to Pugh's method, where the criteria of evaluation in addition to the budget is shown in the next table. The results of evaluation are shown in the following table.

Table 7: Evaluation of three alternatives

Criteria	Weight (of 10)	Alt1	Alt2	Alt3
Availability of the parts	7	+1	+1	+1
User Friendly	10	+1	+1	+1
Robustness	10	+1	-1	-1
User Authentication	4	0	+1	+1
Ease of maintenance	7	0	-1	0
Budget	8	0	+1	0
Total		27	12	12

It is seen that alternative 2 for example has a trade off between robustness and a little improvement in user authentication, and it has the disadvantage that it can't be used directly to extract information as distance and rotation. The third alternative has a weakness in robustness as well as budget. Based on the table, alternative 1 shows the best balance between all the mentioned points with focus

on robustness and accuracy of detection, and thus is chosen as the baseline design.

2.9 MATURING BASELINE DESIGN

After the baseline design was selected, the team proceeds to analyze it and make sure it is mature enough for the next phases. At this point, the team revised the customer's needs, and found out that the required robot doesn't have to follow the user's back, but it can move in front of the user as well, like if the user is pushing it. The team decided to add a new mode of operation called the pushing mode, where the user holds the Aruco marker in the hand and instructs the robot's movement.

Another point was obstacles detection. The team decided not to use sensors as they are not very reliable under different conditions. The use of the camera feed itself is a better solution. This will require a certain type of stereo cameras, and it will simply alert the user that an obstacle is in the way when in following mode. The team took these notes into consideration and updated the design, as it will be shown in the beginning of the next chapter. **The final design is as following:**

Design the robot to follow the user using wheels for movement, based on detecting an Aruco marker that the user will carry on his back, or in his hands. The program will respond only to a certain marker. The camera will detect obstacles in the way and alert the user while in following mode. A phone app will be used for remote control.

CHAPTER – 3 PRODUCT BASELINE DESIGN

3.1 BLOCK DIAGRAM

The following block diagram shows the final high-level block diagram for the baseline design.

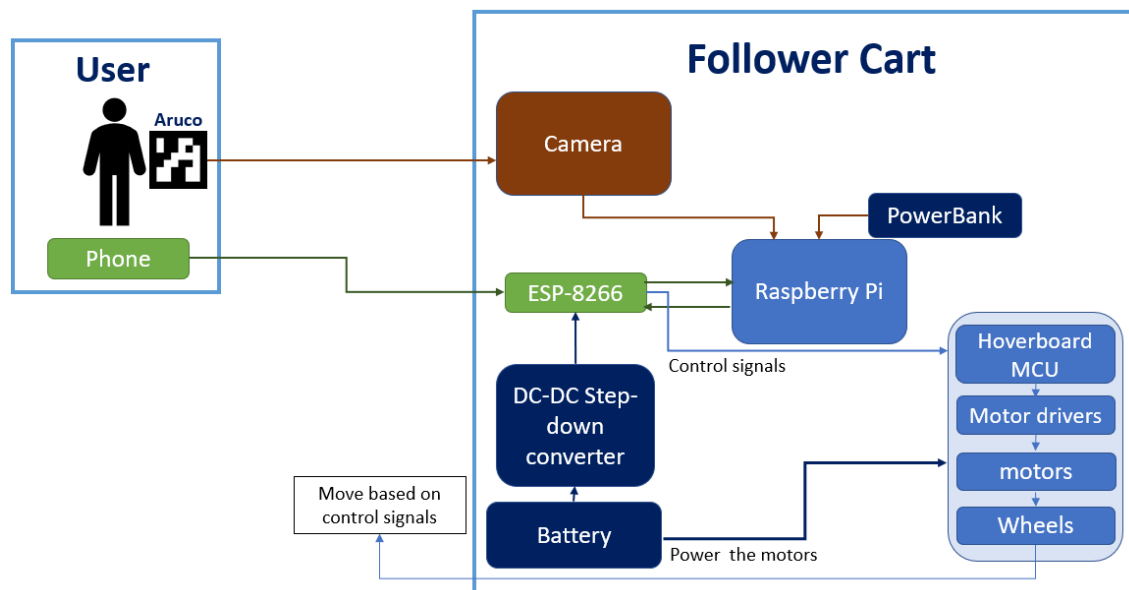


Figure 4: Modified block diagram of baseline design

As it can be seen in the block diagram, the system consists of two major parts: on-board and the user. The user part consists of only the user carrying the Aruco marker, and his phone for remote control.

The on-board block, however, consists of a large system. The main controller is a raspberry pi model B. It receives video input from a camera connected to it, and control signals from a ESP8266-based microcontroller (connected wirelessly to the user's phone). The Raspberry Pi acts based on the given input and sends movement signals to the ESP controller which sends it back to the motor drivers to move or stop.

3.2 SYSTEM DESCRITPION

This section describes different parts of the system. This description is going to be used in the implementation of the project. The section is divided into subsections, where each subsection looks at a different aspect of the design.

3.2.1 Circuit component specifications

The following table shows the specifications of the main electric circuit components used in the circuit.

Table 8: Main electric components specifications

Raspberry Pi 4-RAM 4GB ^[8]	<p>We will be using The Raspberry Pi 4 which has the following specifications:</p> <ul style="list-style-type: none">• Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz• 4GB RAM• 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE Gigabit Ethernet• 2 USB 3.0 ports; 2 USB 2.0 ports.• Raspberry Pi standard 40 pin GPIO header
NodeMCU 8266 ^[9]	<p>We will be using the NodeMCU for wireless connection between the remote controller and Raspberry Pi 4, it will have the following specifications:</p> <ul style="list-style-type: none">• the microcontroller: ESP-8266 32-bit• Wi-Fi Version 802.11 b/g/n• 11 I/O pins
OAK-D Camera	<p>The OAK-D camera is an important part to identify the user and follow him and provide the stereo feed needed to make a disparity map.</p> <p>RGB camera</p> <ul style="list-style-type: none">• Resolution of 12 MP (4056 x 3040 pixels)

	<ul style="list-style-type: none"> • Field of View: Horizontal FOV 68.8 degrees, Diagonal FOV 81 degrees. • 60 fps <p>Stereo camera</p> <ul style="list-style-type: none"> • Synchronized shutter • Resolution 1280 x 800 • Field of View: Horizontal FOV 68.8 degrees, Diagonal FOV 81 degrees. • 120 fps
Battery	<p>The battery is an important used to power the hoverboard and move the AutoCart, it has the following specifications:</p> <ul style="list-style-type: none"> • Capacity: 4.4 AH • output current (A): 15-20 • power (watt-hour): 158.4, • rated input voltage: 100-240V ~, 50 / 60 hz • charging time 1.5-2 hours.

3.2.2 Flowcharts for software blocks

In the AutoCart, the Raspberry Pi is the mind or the main controller of all actions. The software being run on the Raspberry Pi is shown in simplified version in the following flowchart.

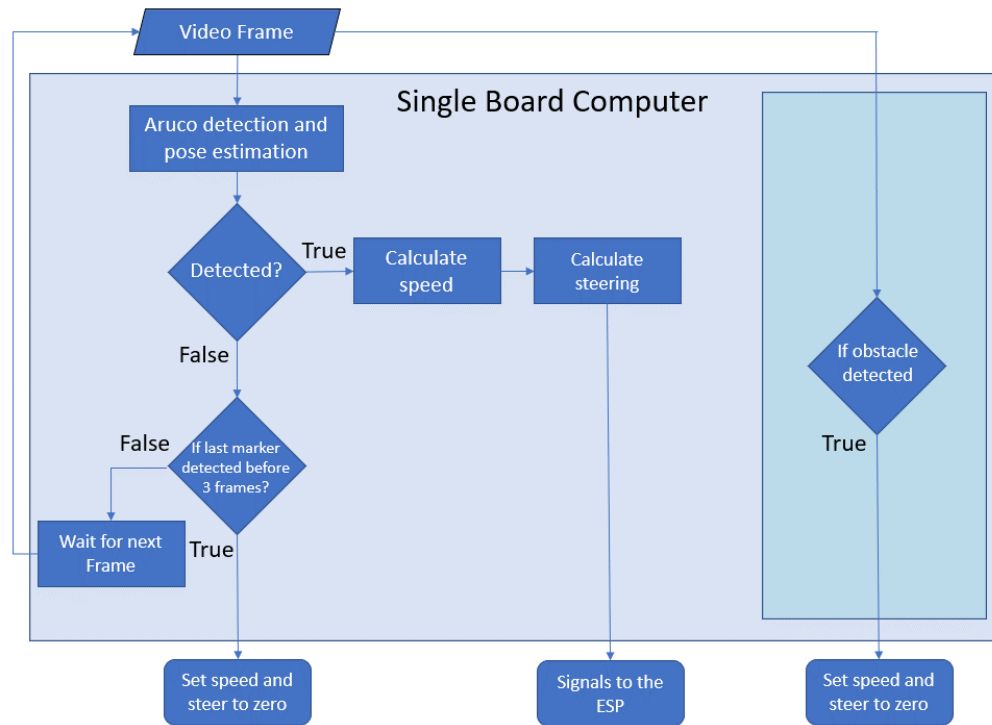


Figure 5: Simplified software flowchart

3.2.3 Mechanical specifications of the case

The main physical components we will use in our project are a cart made of wood and aluminum and a Hoverboard. The width of the hoverboard (wheel to wheel) is 45 cm. The cart used as a mechanical structure for the robot is made of aluminum and wood. It has the width of 85cm, length of 55cm, and height of 88cm. The structure has multiple levels to put stuff on, giving it more space for the user to transit his things. The next figure shows the dimensions and 2D drawing of the structure.

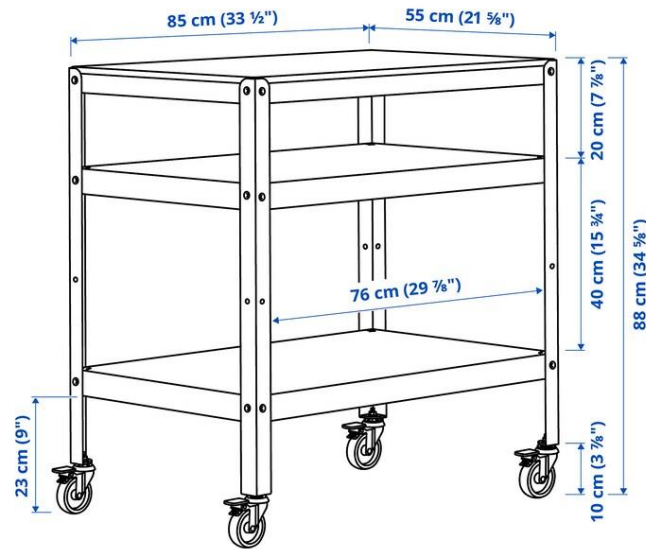


Figure 6: The mechanical structure of the project as provided by the manufacturer^[10]

3.2.4 Possible aesthetics

An important part of making a product is creating something that will attract buyers only by looking at. Since the AutoCart is supposed to be used in closed environments which may include houses, or office buildings, we made sure to make it looking good for these environments. The selection of the mechanical structure to was done to blend well with these environments.



Figure 7: Mechanical structure (raw before modifications)

Another part of this task was to use 3D printing to make compartments for any attachments needed in the project. For example, the team printed a camera stand, a battery compartment, and a compartment for both the ESP board and the Raspberry Pi board, and made sure to make the big components to be hidden. All of these parts are shown in the following images:



Figure 8: From left to right: Raspberry Pi compartment, camera stand, and ESP compartment

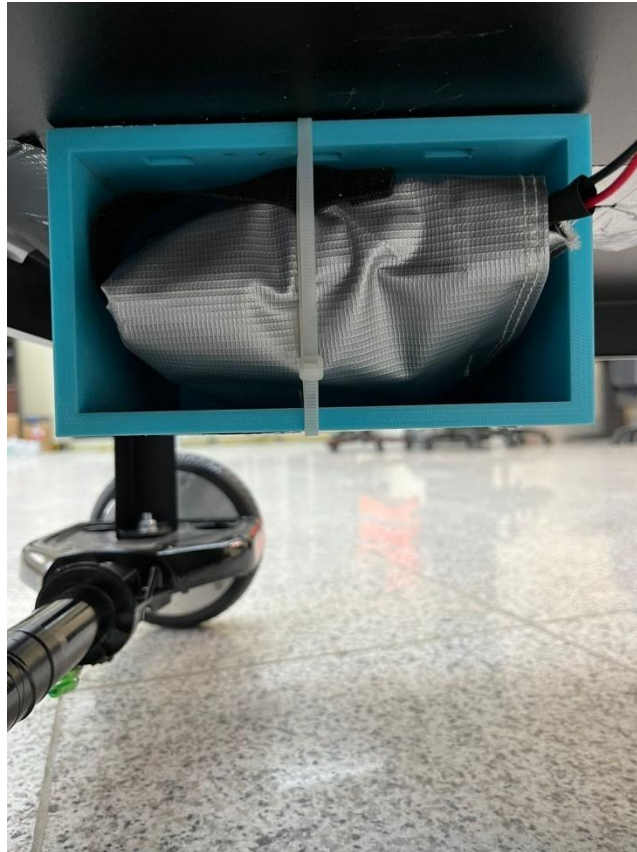


Figure 9: The battery in its compartment, hidden under the mechanical structure

3.2.5 Operating Instructions

These operation instructions are to be used by the user of AutoCart:

1. Put your items/loads in the AutoCart cargo space.
2. Press the power button on the AutoCart.
 - A. Remote control mode:
 - I. On your phone, connect to the network named “KAU_Office_Delivery_Robot”. Provide the password.
 - II. After connecting, run the remote control software provided.
 - III. Use the tools on screen to move the robot forward, backward, steer left and right, and see the feedback printed on screen including battery voltage.
 - B. Following mode:
 - I. Make sure the Aruco marker is clear on your back.

- II. Press the button that runs the Raspberry Pi, and wait for it to boot.
 - III. After booting, the program will automatically run and you will hear a sound from the camera indicating that.
 - IV. Stand with your back facing the Camera.
 - V. Start moving, and the robot will start following you. Keep an eye on the road and make sure that the robot is following you and that you are not out of its sight.
 - VI. To change to pushing mode, turn around such that the Aruco marker is hidden to the camera behind your back. Use the mode button to change to push mode. Then use the hand held Aruco marker in front of the camera, the robot will start moving backward like if you are pushing it.
3. To stop the AutoCart in either mode, press the power button for the Raspberry Pi if it was on, and then the power button of the hoverboard.

CHAPTER – 4 IMPLEMENTATION

4.1 MECHANICAL MODIFICATIONS

The team decided to use an electric hoverboard as a base of movement for the project. However, it was to be attached to the mechanical structure in a reliable way. The team sought help from KAU engineers in the mechanical engineering department. A challenge appeared as the width of the hoverboard was smaller than the width of the mechanical structure of the project.



Figure 10: Comparing hoverboard width with mechanical structure width

The mechanical engineers suggested to remove the supporting metal rod that connects the two parts of the hoverboard and replace it with a longer 35mm aluminum alloy rod. See the next image for part of the process.



Figure 11: An image after disassembling the two parts of the hoverboard showing the small rod to be replaced.

Another problem that faced the team, was how to attach the hoverboard to the structure itself. The team wanted to do welding. However, the mechanical engineers suggested that the materials of the table and the hoverboard are different, so welding cannot be achieved because of the temperature difference. The other solution was to drill the base of the hoverboard, and attach it with 8mm screws and bolts. The following image shows the structure after the attachment was successful.



Figure 12: An image of the structure after attaching the hoverboard

4.2 COMMUNICATION AND HOVERBOARD CONTROL

Our project main mechanical system is a Hoverboard or known as a balancing scooter. The purpose is to use it as the driving motor of our project. The structure of communication will include three main parts, The embedded controller in the hoverboard that is used to control its motors and balancing system. In order control the hoverboard's motors wirelessly to achieve our project's goals, we used ESP-8266 chip, which is a microcontroller that provides Wi-Fi communication. The final part is a control program, which connects to the ESP model and communicates to it the proper control signals according to the implemented algorithm (i.e., following algorithm, or remote-control application).

4.2.1 First trial

The team first had to modify the mainboard brain of the hoverboard, which is an STM32 board, to accept our input. Using Some tutorials on the internet (See [11]), the team was able to modify the hoverboard's controller to be controlled with our ESP board.

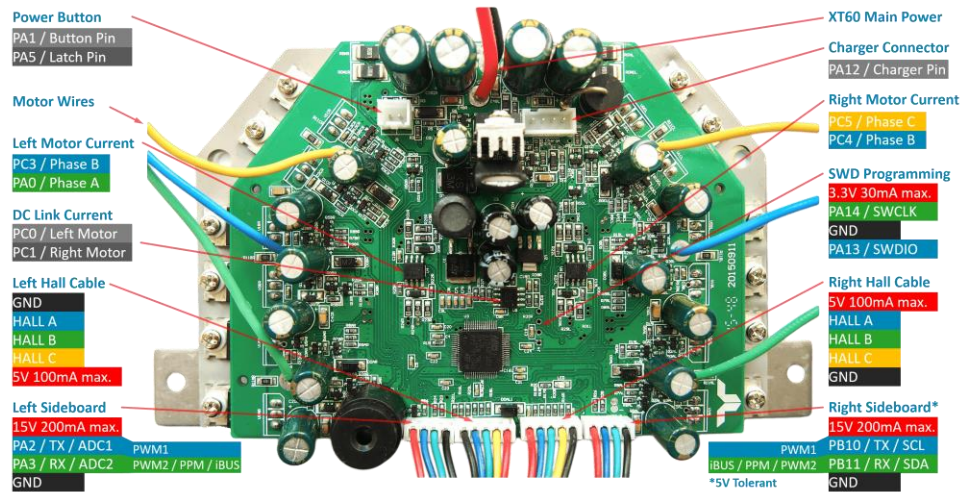


Figure 13: Hoverboard mainboard^[11]

The team flashed a C++ code to the ESP chip with the help of the advisor, that accepts TCP connection, and used a simple MATLAB code to test the ESP Wi-Fi connection through a computer. The MATLAB code's purpose was to only test the connection and perform basic movements by manually passing speed and steer signals. In our first attempt to run the system we ran into an error when connecting to the ESP-8266 Wi-Fi channel, so we assumed there is a problem within the ESP code, we extensively debugged the C++ code to solve the issue. However, we found no issues within the code itself.

4.2.2 Second trial

We assumed the problem is caused by a production fault in the ESP-8266 we had, so we tested another chip of the same model, and that was the case. we successfully connected our laptop to the ESP board, with one additional problem. The connection disconnects immediately after the connection. The team took more time to reverse engineer the MATLAB code, and finally understood the fact that in TCP communication, send and receive functions have to come in same order in both the server and the client. The team fixed the issue in the order of send/receive functions, and the connection was successful. The next step was converting the code to python to be able to integrate it with the rest of the software.

Finally, to integrate the ESP board with the hoverboard, there was a challenge as the ESP works on only 3.3V while the hoverboard provides 5V. So there was a need to make a simple step-down converter circuit with the hoverboard as following:

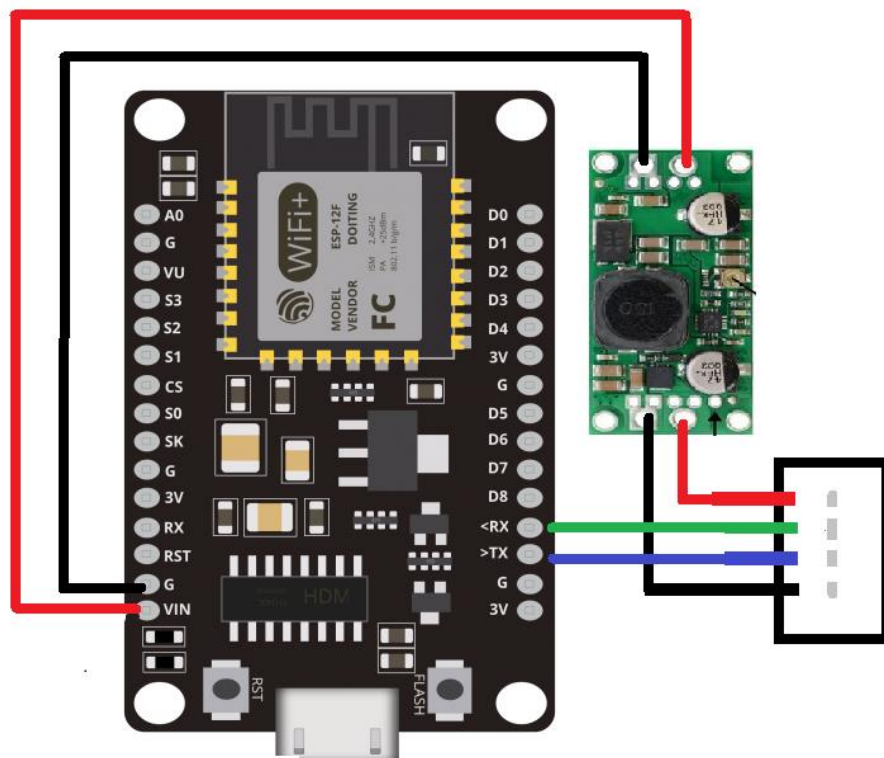


Figure 14: ESP-8266 Circuit

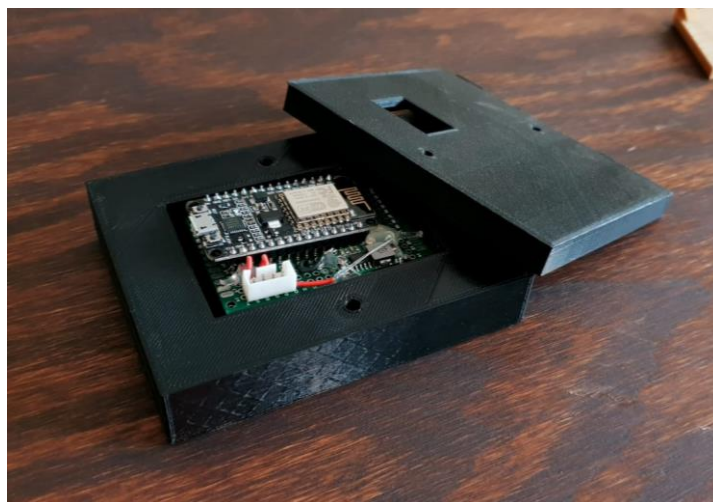


Figure 15: ESP-8266 After soldering

4.3 CAMERA SETUP

4.3.1 First trail (Web cam)

At the start of our implementation, we used a USB Webcam for a limited demonstration. We used the Webcam for some time. However, as the project matured, we realized we needed a stereo-camera with the capability of providing a disparity map (indicates how far or close objects are from the cam using two lens).



Figure 16: Webcam used at the start of the project

4.3.2 Second trail (Bi-cam)

The team brought a stereo camera with CSI connections and worked on its setup for a time. This cam had many issues one of which is its limited documentation.



Figure 17: Stereo Camera and its output

At this phase of the project, we were using Jetson Nano single board computer which was compatible with the camera. After facing many issues and challenges we consulted our advisor and decided to try another camera and a different single board computer.

4.3.1 Third trail (OAK-D)



Figure 18: OAK-D Camera

Finally, the team started using the OAK-D camera. This product provides both a stereo-cam and a normal cam located at its center. The stereo part allows us to make a disparity map in our project to detect obstacles.

4.4 USER DETECTION USING ARUCO MARKER

4.4.1 First trial

The first step in Aruco detection is to calibrate the camera. Calibration process is done by printing a checkboard on an A4 paper, then taking multiple pictures of different positions to it, and passing it to a calibration function provided in the opencv library for python. The team did 30 images, and this returned two arrays called camera matrix and distortion coefficients, which are used in the Aruco pose estimation process discussed later.



Figure 19: Example from the calibration process

The next step is to detect an Aruco marker in screen. There are many “dictionaries” for Aruco markers, so the team chose a dictionary called (DICT_6X6_250) which contains 250 different markers to be detected. With a camera set-up in Python, one has to pass the camera frames to an opencv-contrib-python function called “detectMarkers”, which will search the frame for any Aruco markers of the specified dictionary, and return their screen position and their id.

The next step, which is the most important step, is to apply pose estimation, which means to estimate the exact position and orientation of the marker in real world, from the video stream. The opencv library returns two vectors as a result of pose estimation:

1- Translation vector: this one contains 3 values that represent the distances of the marker from the camera, in the 3 axis. It is as following:

- a- The x axis shows the distance of the marker to the left and right of the camera
- b- The y-axis shows the distance up and down of the camera level.
- c- The z-axis shows how close and far the marker is from the camera.

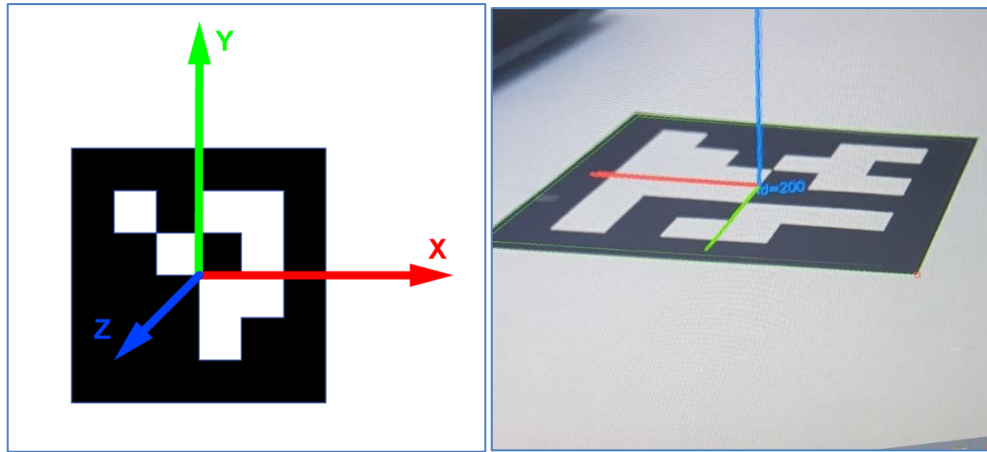


Figure 20: The axis of Aruco marker

2- Rotation vector: this also has 3 axis, but the raw data can't be used directly and require more processing to be converted to Euler angles. The result is the rotation angles of the Aruco marker around the 3 axis.

The team followed the available tutorials from the documentation(see [12]), but the results would always show +30cm difference, which is unacceptable for our purpose.

4.4.2 Second trial

The team tried multiple fixes to solve the pose estimation issue. One of the challenges is that the documentation is written in C++, but the same functions apply to Python. The documentation also didn't had all the information provided in one place which lead the team to look for multiple resources with no clue. One of the attempts was recalibration, which had a small impact on the results. Another attempt was to rewrite the code separately and use it on another camera, which at that time, was an OAK-D camera. This solved the issue at the time, but when reattached the code to the main file, the problem showed up again.

4.4.3 Third trial

At this stage, it was clear that the problem isn't in calibration or the camera, but in the code itself. After precise debugging, the team found out that we were “**resizing**” the camera frame to be able to show it on screen. However, resizing it means that the computer will see different dimensions and the measurements of position will get resized accordingly. The team moved the frame resize to be after

applying pose estimation, and the results became accurate with less than 1cm error.

4.5 FOLLOWING ALGORITHMS

4.5.1 First trial

There are three modes of operation for the project.

- 1- Remote control: this one is implemented as a phone app.
- 2- Pushing: the user holds the marker in his hand and shows it to the camera, the robot moves away from the marker.
- 3- Following: the marker is at the back of the user. The user walks while the camera detects the marker and the robot follows it.

Where both pushing and following are actually considered as “follow” by the robot. The goal is to design a way to make the Aruco marker control the movement of the robot. The team started with the following algorithm. The algorithm simply needs to find the distance between the user and the camera, and whether the user is at left or right, then it sends an appropriate control signal to the hoverboard. At start, the team used a very straight forward algorithm:

- 1- If the user is at less than 100cm:
 - a. Stop
- 2- If the user is far more than 100cm:
 - a. Move the hoverboard with constant speed until the distance is 100cm

The problem with such algorithm is that users walk with different speeds, and fixing the speed of the hoverboard will cause irritation to the user.

The team also used the rotation of the Aruco marker to determine when the hoverboard should head to the left, center, or right when moving. However, this was not very ideal as the user might move his back without actually turning, which will cause false steering for the hoverboard and leads to the user being out of the line of sight. Another algorithm needed to be implemented.

4.5.2 Second trial

Starting with the speed of movement, the team needed a better way to assign speed to the robot, in an adaptive way:

- 1- Set a maximum distance that the robot will not follow the user if the user goes beyond it.
- 2- Set the minimum distance required by the robot to start moving forward. In our case, this is 100cm
- 3- Set the maximum distance at which the robot stops moving, and the maximum speed the robot should ever reach.
- 4- Divide the current user's distance by the maximum distance. This gives a fraction that grows whenever the user goes far

$$\text{Distance fraction} = \frac{\text{current distance}}{\text{max distance}}$$

- 5- The current speed can be obtained by multiply the fraction by the maximum possible speed.

$$\text{speed} = \text{distance fraction} \times \text{max speed}$$

This way, the robot will get faster only if the user starts moving faster. The team tested the algorithm and it ran smoothly, reducing the effects of the previous problems.

For the steering part, to determine if the user is turning left or right, the team designed a different algorithm, depending on whether the "position" of the aruco marker is at the left, center, or right of the screen. However, this shows a problem: when do we consider the user to be at the "right" of the screen, and when do we consider him near the center? Using the raw x-axis distance is not feasible. Look at the following graph for an explanation:

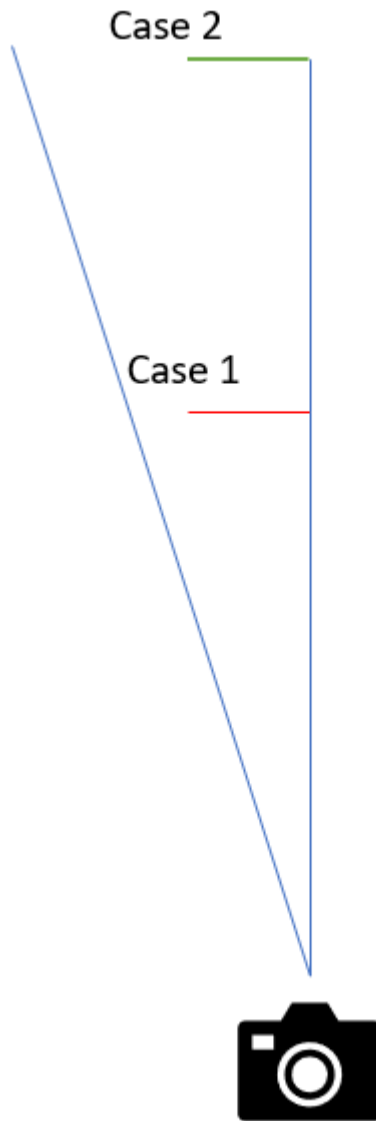


Figure 21: Explaining the field of view of the camera

Notice in the previous figure that the blue lines show the field of view of the camera. If the user is standing at case 1 (the edge of the red line) notice that he is very close to the edge of field of view of the camera, and so the robot has to take a sharp turn to keep him in the frame. However, if the user is at case 2 (the edge of the green line) the user is still considered near the center and the robot needs a slight steering only. The important note here is, both case 1 and case 2 have the **same distance** to the left! So, if we use the raw distance, how can we detect if we need a little or sharp turn?

The algorithm the team used depends on converting the x-axis distance into a fraction or a percentage, is as following:

- 1- Obtain the field of view of the camera (69 degrees for Aruco).

2- We derive the Pythagoras formula:

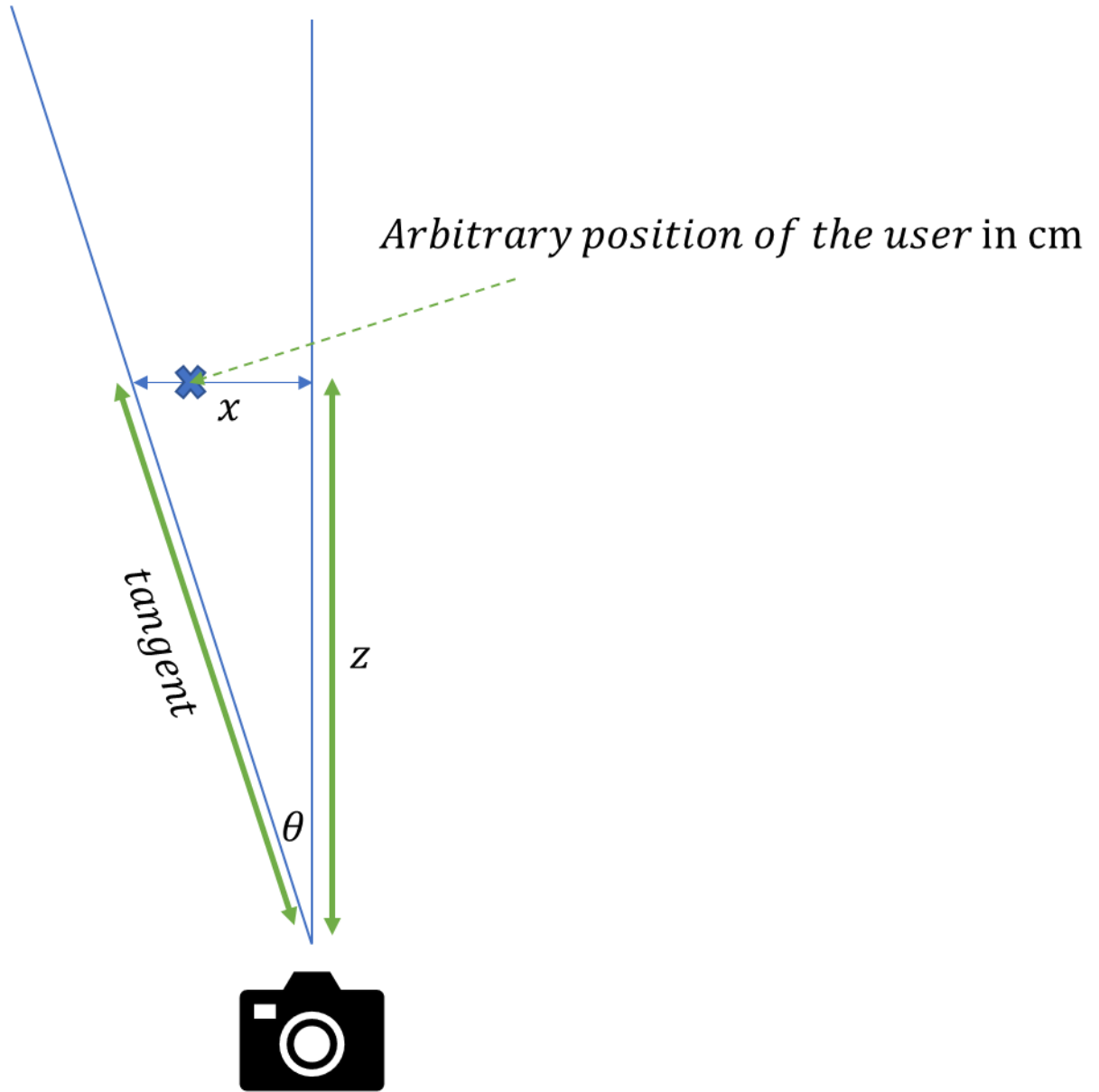


Figure 22: Explaining steering algorithm

3- The goal is to calculate x , given the values of θ and z . First, we calculate the tangent as:

$$tangent = \frac{z}{\cos \theta}$$

Then, we can calculate x as:

$$x = \sqrt{tangent^2 - z^2}$$

4- Finally, we can convert the position of the user into a fraction or a percentage (dividing the x -axis position by the calculated value of x).

$$x_{percentage} = \frac{user's\ x\ position}{x} \times 100\%$$

This way, we can say as an example, that if the user is at more than 75% of the field of view, we need a sharper turn, and this will work at any distance the user is at.

However, this algorithm for steering is good in the following mode only. In the pushing mode, this is not very reliable, as the user needs to move his whole hand to the left and right when he needs to take a turn. The team needed another algorithm for that purpose.

4.5.3 Third trial

For pushing mode, we use the same algorithm for determining the speed, with the difference that the speed increases if the user gets closer, since he is imitating a “push” behavior. However, for the steering, we decided to use the Aruco as a steering wheel. This means that the user can turn the marker in his hand like a steering wheel (tilting it left and right), and the cart will turn left and right. This makes the control easier in the pushing mode.

4.6 THE TRANSITION TO SINGLE-BOARD COMPUTER

Through the development of the main code, we used laptops to write, test and debug our code. That is to benefit from faster execution times and the many tools provided to ease debugging codes such as modern IDEs, (i.e. Pycharm community). After we fully developed and tested our code, we planned to move it into a smaller single-board computer, which can easily fit within the compartment on the robot and consumes much less power than a laptop.

Since we are using Python, which is an interpretive high-level programming language that can run on any system if installed correctly, we assumed to have no trouble to transition from a laptop to a single-board computer. Our program required dependencies to run the depth camera from USB and required installing OpenCV and DepthAi libraries on a Python virtual environment. So, in order to transition successfully we had to properly setup our single-board computer to run the code.

4.6.1 First trial: Raspberry Pi 3

The team initially used a Raspberry Pi 3 model-B of 1GB RAM, which we assumed to be sufficient for our program. We installed Raspbian OS -Desktop on Raspberry and created a Python Virtual environment. However, we ran into a problem when installing OpenCV-contrib-python library, which is a crucial library to process Aruco markers. The installing runs until the building phase, where it is processing for a very long and ends up displaying a timeout error. We looked for solutions to this problem.

A solution we tried was to download the official OpenCV files and manually building them instead of using Python's "pip install" command. however, the building operation was RAM-dependent and took so much time even with increased swap size, which also ended in a timeout error.

Another solution was to install a previous and lighter version of the OpenCV library. we tried this solution by installing the version "OpenCV==4.5.3.56", this solution took some time to install but it worked in the end. we successfully ran our code, but the performance of frames processing was underwhelming. The code was processing camera's output very low frames per second without applying the algorithms. Applying the Aruco detection algorithm made it even slower to the extent of multiple seconds of delay!

4.6.2 Second trial: solving undervoltage problem

The performance results of the first trial were very inconvenient, so we analyzed the outputs and measured the performance of the system, we realized that powering Raspberry Pi 3 with a power-bank was not delivering sufficient power, which caused a much-reduced performance. So, we tested the raspberry Pi dependently using an official powering adapter, however the problem persisted even with proper powering. after taking advice from our advisor and checking different alternatives we decided to replace the Raspberry Pi 3 of 1GB RAM with a newer Raspberry Pi 4 – 4GB RAM.

4.6.3 Third trial: transitioning to Raspberry Pi 4

At first, we setup the new Raspberry Pi 4 with the same Raspbian Desktop-OS. We ran into a problem where the Raspberry could not detect any nearby Wi-fi channels, which is a major flaw. The team looked into different solutions. one

solution proposed there might be an error during the installation of the operating system. Since Raspberry Pi model 4 supports different operating systems, we tried Ubuntu Desktop 22.04 LTS. This time we had no issues detecting Wi-Fi channels. we proceeded to create a Python virtual environment and install the needed dependencies and Python libraries. The team ran the code, and it worked flawlessly yielding around 26-30 frames per second, which is very sufficient for our detecting and following algorithm.

4.7 USER INTERFACE

One part of our project is developing a phone application for user interface and the remote-control mode. Our approach was to design something minimal and does not take as much effort since our main focus of the project is delivering a robot that is controlled by algorithms that process the camera output. However, the goal of the android app is to deliver maximum user satisfaction and convivence. the app will feature two sliders, control and speed, which are used to control the movement of the cart, it will also display power levels. We have chosen to develop our application on Android OS because it allows customizability and a lot of support for development.

4.7.1 First trial

In our first attempt we have received a controller app in “Android package file” or APK from our advisor to test with it, we successfully installed and ran the application on an android emulator running on a Windows 10 OS computer host. later on, when we brought an android device specific to the project, we ran into a problem when installing APK file, the issue required a lot of effort, so we started to develop our own simple app.

4.7.2 Second trial

Since we did not have much expertise with android studio or android development, we decided to use a simpler option to build the app as a simple structure. The structure is based on Simulink Android package, which a customizable development environment. We modified the limits of speed integers and added few widgets, and feedback information that shows the battery voltage to the user,

satisfying one of the musts, and successfully built the program on our Android mobile phone.

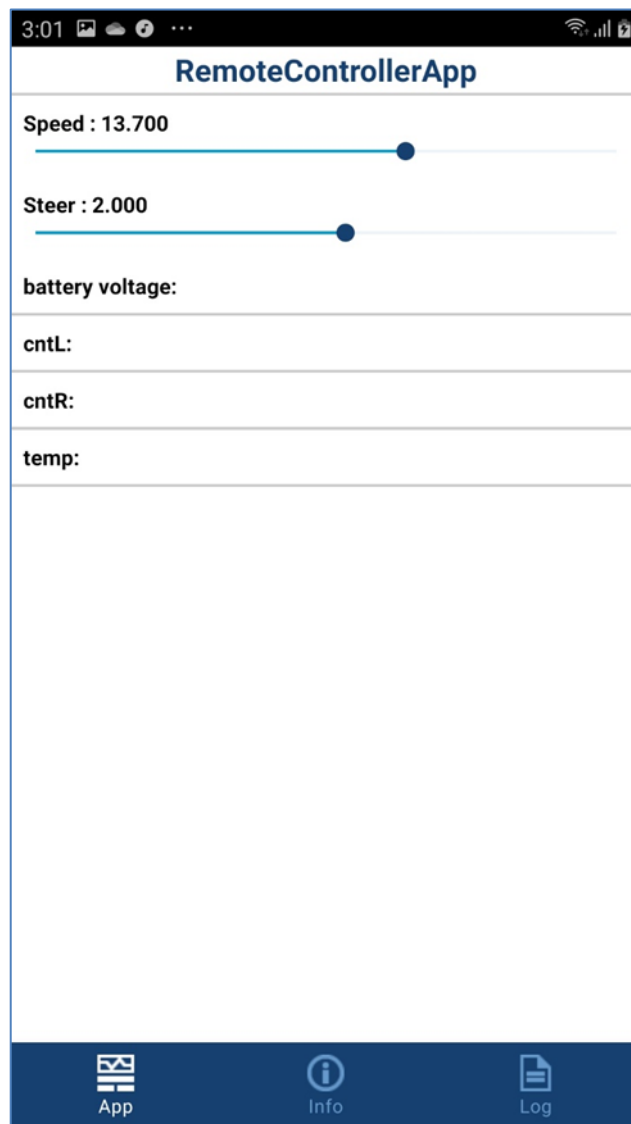


Figure 23: User Interface

4.8 3D PRINTING

One part in finalizing the project was to prepare some compartments for some components of the project. This was done using 3D printing. The first piece to print was the ESP8266 board compartment. However, because of the lack of experience, the team at first printed a missed dimensions in the housing. The problem was later solved as in Figure 15. Another part the team failed at printing at start was the camera stand. The first version was printed with PLA material, and it was so easy to break that it didn't hold the camera correctly.



Figure 24: First camera stand – The final camera stand

Later, the team used the modeling software to increase the thickness of weak sections and reinforced some parts. The material also changed from PLA to ABS material which is mechanically stronger than PLA.

4.9 FINAL PRODUCT

After implementing the project and making sure all the components are working, the project looks like the following:

- 1- The general structure



Figure 25: The final project - general view

The general structure was chosen, as mentioned before, to make the project look appealing and to blend well in office environments. The wooden top and the black body play that role. Also, the camera on its stand looks clear so that the user knows where to point the Aruco marker. Most of the cables were hidden, except for the camera cable, which can't be hidden.

2- The Raspberry Pi and its battery fixed

The Raspberry Pi has been hidden under the wooden surface so that it doesn't get damaged by any liquids or mishandling. The battery was affixed next to it to limit the length of wiring needed.



Figure 26: Final project - Raspberry Pi

3- The hoverboard battery fixed

The battery was fixed under the body of the cart so that it is hidden away and safe from any mishandling. To ensure safety, it was placed in a special encloser as discussed later in Chapter 5. See Figure 9.

4- The suit with an Aruco marker

To enable the user to use the robot, we supply the product with a suit, that has an Aruco marker attached to it on the back. The suit is very simple to make sure it doesn't irritate the user when wearing it. The team also provides a handheld version of the marker to be used in pushing mode.



Figure 27: One of the team members showing the suit

CHAPTER – 5 RESULTS, DISCUSSION, AND CONCLUSIONS

In our project, there are 3 main parts that we tested and validated: the hoverboard, the Aruco detection and pose estimation, and the communication. The following subsection discusses the results of our validation experiments, which can be found in details, attached in appendix A.

5.1 RESULTS AND DISCUSSION

5.1.1 Aruco detection accuracy results

To make sure of reliability of our system, we tried to validate the amount of error in distance and rotation estimation. So, for the first part to validate the distance, a measure tape was spread on the ground, where its tip is parallel with the tip of the camera lens. Then, the Aruco marker was placed at different places on the tape, and its distance from the camera according to the tape was collected, in addition to the distance that the our software estimated, The results were put in tables, and the error was calculated and graphed, as following:

Table 9: Results of distance estimation error

True Value (cm)	Average measured distance (cm)	error(%)	error(cm)
30	30.2944	0.9815	0.2944
35	35.5225	1.4928	0.5225
40	40.5953	1.4882	0.5953
45	45.5639	1.2532	0.5639
50	50.6696	1.3392	0.6696
55	55.7511	1.3657	0.7511
60	60.8023	1.3372	0.8023
65	65.6659	1.0245	0.6659

70	70.7653	1.0932	0.7653
75	75.7175	0.9567	0.7175
80	80.7274	0.9092	0.7274
85	85.3647	0.4290	0.3647
90	89.5061	0.5488	0.4939
95	94.2296	0.8109	0.7704
100	101.0551	1.0551	1.0551
120	119.1417	0.7152	0.8583
140	141.2184	0.8703	1.2184
160	161.1778	0.7361	1.1778
180	180.5290	0.2939	0.5290
200	200.3752	0.1876	0.3752
230	232.9995	1.3041	2.9995
260	265.2293	2.0113	5.2293
290	296.3772	2.1990	6.3772
300	307.4715	2.4905	7.4715

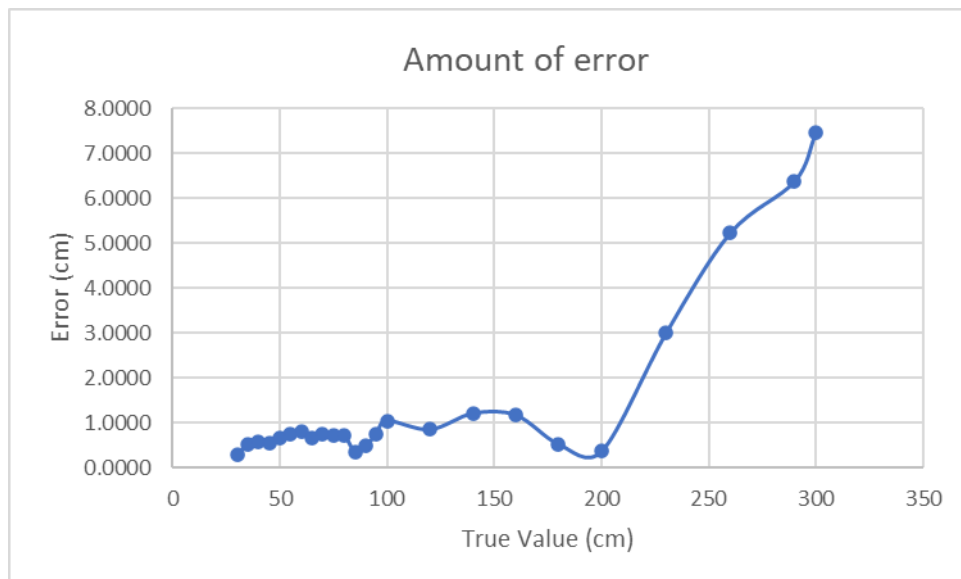


Figure 28: Plot of the error in distance estimation

By inspecting the distance error graph, we notice that error in cm is less than 1cm for most of the data under 200cm. The little variations might be due to human

reading errors, and lightning conditions. The way this distance is being used is that we will move forward at certain threshold, which is usually 100cm. Thus, 1cm error is very acceptable and will not cause accidents, since the cart is already 100cm away from the user, so 1 cm error at that distance is safe. We notice that the error increases as the camera goes further away from the marker. However, this is not expected to affect the working of the project, since the final product will not be that far from the user, as specified previously.

The second part of the experiment is to calculate the error in rotation angle estimation. For this purpose, the team modified the software a little bit to show some alignment lines on the screen that can be used as angle indicators. Then, the marker was placed on a flat surface and rotated, while aligning it with the lines, and the angle was collected, from both the alignment lines (real value) and the software estimation. The results were put in tables and plotted as following:

Table 10: Results of "rotation angle" estimation error

True value (degrees)	Average Measured angle (degrees)	Error (%)	Error (Degree)
45	45.15935	0.354115	0.159352
90	89.80323	0.218631	0.196768
135	135.1986	0.147123	0.198615
180	180.1367	0.075917	0.136651
225	225.2762	0.122736	0.276157
270	270.1686	0.062428	0.168555
315	314.8087	0.060717	0.191258
360	359.7500	0.069454	0.250034

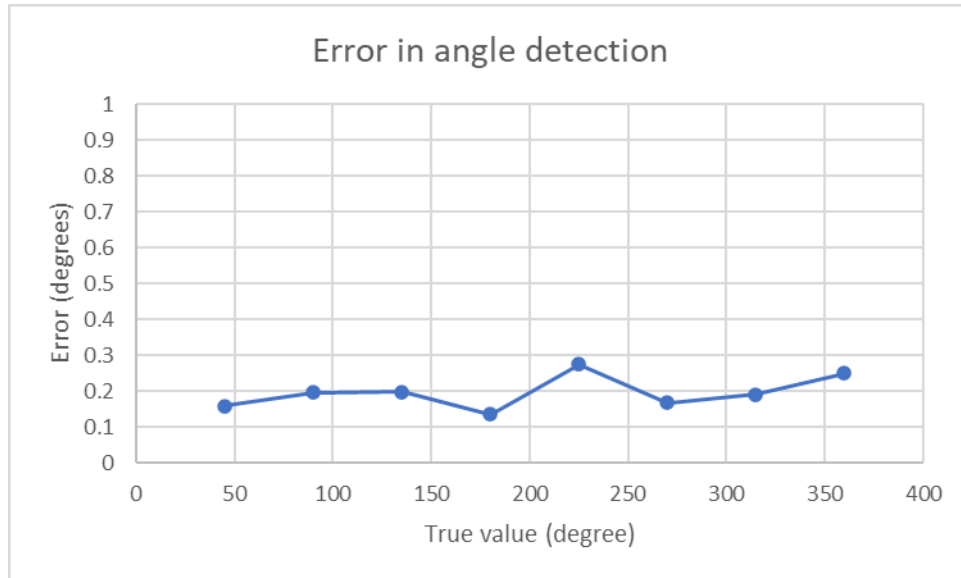


Figure 29: plot of "rotation angle" estimation error

We can see that the results are consistent for the most part, but there are some variations at 225 degrees. This is usually due to the human reading errors and the method we used in reading the values. However, all of the data show less than 0.3 degrees of error, which is very acceptable. However, it is wise to add a little margin of error in our steering algorithm in the following mode.

5.1.2 Hoverboard speed results under different weights

In this subsection, we show the results we obtained when validating the ability of the hoverboard to provide consistent speed under different weights. This is crucial for the project, since we need the robot to be able to keep up with the speed of the user. The team used exercising weights for this purpose. The robot was set to move in a straight line, and its speed was set inside the code to 0.46m/s. Next, the team started increasing the weight in each iteration and calculating its real-world speed, to compare it with the code speed. 3 readings were taken for each weight, and the average turned out as the following:



Figure 30: Average speeds under different loads

The results show some minor fluctuations in speed. Also, we can see that the intended speed (0.46m/s) was never reached. This problem could be due to the fact that the hoverboards are designed to have the weight centered on top of it. However, in our case, the weight is scattered around wide area. The error does not exceed 4.3% which is roughly 0.02m/s, that is, 2cm/s, which doesn't have a high impact on our solution. In addition, the algorithm was designed in such a way that the speed increases whenever the distance is increased, which will make up for the slight speed error.

5.1.3 Communication results

In this subsection we show the results of the conducted communication validation Test. The test examines the strength of the signal, and the test is conducted with varying the range and the environment from an open space to a narrow corridor. These are the data collected by varying the distance, in two different environments.

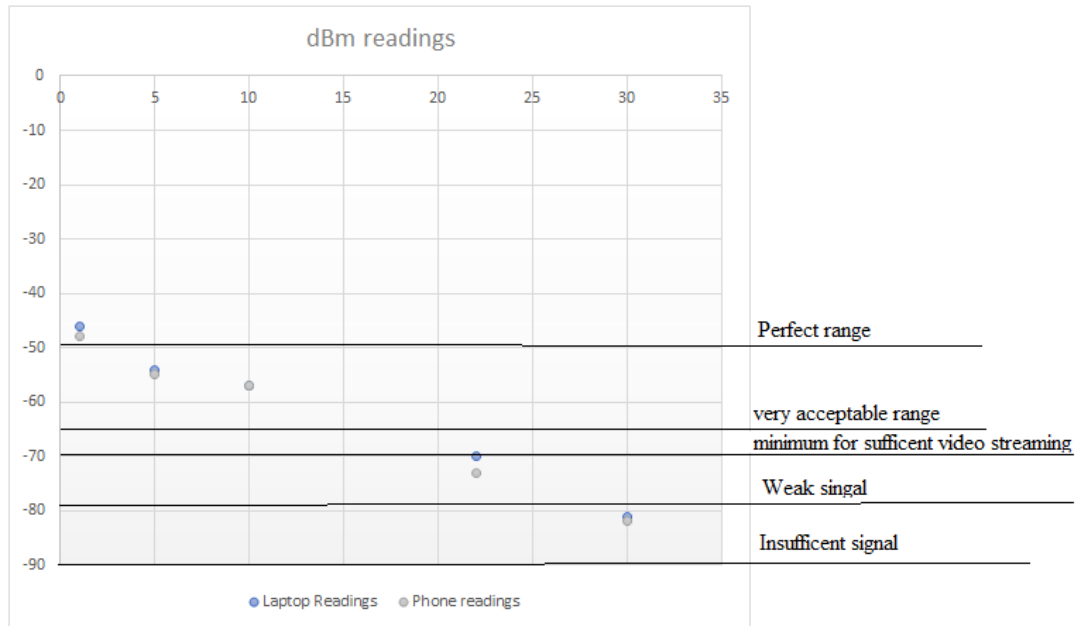


Figure 31: Data collection of communication testing

The y-axis shows the signal strength while the x-axis shows the distance, we can clearly see how the signal strength weakens with the increase of distance which is per assumed in the Validation process (see appendix A). The difference between narrow corridors and open space was not very significant, this provides more confidence in our project. By inspecting the above figure, we see that safe range for consistent signal is within the range 0-20 meters, which is a very acceptable range for closed environments.

During this test we had to keep up the connection for a long time to perform the test over various ranges and environments, the connection was stable during the test, and we had no trouble with communicating with send and receive signals in ranges from 0 to 25 meters. These results prove the integrity of the communication system we implemented in our project.

5.1.4 Results of testing after final assembly

The goal of this experiment was to prove the ability of the project to follow the user between two points, while carrying some weights. The experiment was done by marking the floor with 3 markers, with 10m distance between each two. Then, the operator stands at the first marker and positions the AutoCart behind him with 1m. The operator then, with the Aruco marker on his back, starts moving to the next floor marker, and records the time he reached his destination, and the time it took the robot to follow him to the same destination. The difference between the two timings (the delay between the user reaching the destination and the robot reaching the destination following the user) was calculated, and data collected several times, for both 10m and 20m. The results were discussed in appendix A, and are shown in the following graph:

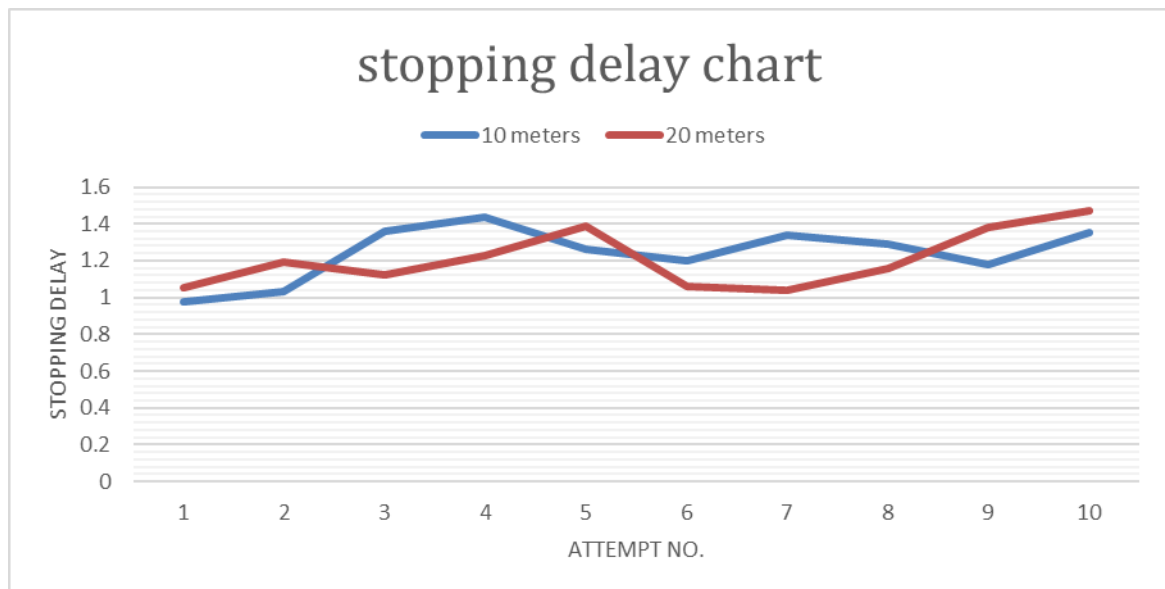


Figure 32: Testing the product, stopping delay chart

from the chart and data tables we can see that the maximum delay was 1.47 seconds and lowest delay was 0.98s, and the standard deviation of the stopping delays is 0.148, meaning that the spread of the values of stopping delays has been not very high. this goes well with our aim to deliver smooth movement with not much of unpredictable and sudden changes in speed. This also proved the capability of the finished product to transport object between two indoor points.

5.2 EVALUATION OF SOLUTIONS

In this section, you should reflect upon the final product based on the results you discussed earlier. Reflection should be from different technical and global aspects, for example:

5.2.1 *Technical Aspects*

By looking at the results of the validation experiments, we can confirm that the project can satisfy the customer needs. The project has been validated to support more than 70kg of weight, and is able to transport it at consistent speeds, and will keep up with the speed of the user when in following mode, while keeping a safe distance from the user. The pushing mode is also available, in addition to the remote-controlled mode. Also, the battery used for hoverboard is designed for this kind of devices to work more than 2 hours per charge, which satisfies the customer needs.

However, this doesn't mean that the solution is perfect. The team was expecting that the project will have difficulties to deal with corners and turns while in following mode, and that's why it was stated previously as a limitation of the design. The robot requires the user to take slow turns, while keeping his back within the line of sight of the camera, in order for the robot to be able to follow through corners and turns. Sharp turns will result in immediate lose of sight for the camera. This is partially acceptable since the robot is only semi-autonomous and not designed as fully autonomous, so it requires the attention of the user. To solve this problem, the team provided the pushing mode, which gives an easier method to steer the robot left and right. A future work could use multiple Aruco markers on multiple sides of the body to make making turns smoother and more convenient, and to overcome this limitation. Also, it might be possible to port the project from a single board computer to a microcontroller, but this will require porting the processing of camera frames into the camera itself, which might need to use more sophisticated cameras.

5.2.2 Environmental Impacts

In our project, we are using a 36V Lithium-ion battery to power the hoverboard, and 12V Lead-acid battery to power the Raspberry Pi. These batteries are rechargeable and for that are expected to live with the users for time. However, as soon as they become old and useless, these batteries will have bad effect on the environment due to their toxic chemicals. The disposal of these batteries should be carried by specialized professional companies.^[13]

On another aspect, the general mechanical structure is made of aluminum and wood. While wood and aluminum are environment friendly, the plastic in the hoverboard isn't, and so it is wise to put such a project into recycling process when it is to be disposed. The team tried to minimize the use of plastic, which is why the team used PLA, a more environment friendly material in 3D printing some of the parts in the project.^[14]

5.2.3 Safety Aspects

5.2.3.1 Chemical and fire risk

The Lithium-ion batteries produce chemical risk in case they were handled incorrectly. This might include explosions, fire, or toxic smokes. To make the project safe for the users, the team used a special encloser called "LIPO Guard" for this kind of batteries. This will help prevent it from having explosions and will make it harder to be mishandled.^[15]

5.2.3.2 Electrical risk

While most of the high current wires are inside the hoverboard plastic encloser, there was still some holes in the body. The team made sure to close any possible hole that can lead water and other liquids to any of the internal circuits. The team also 3D printed some non-conductive components to be used to fix the battery and other small electronics, to prevent any possible electric shocks for the user.

5.2.3.3 Mechanical Energy Risk

Since the project is designed to have a safe distance from the user, it is considered safe accident-wise. It also uses disparity map to detect any sudden obstacles in the way and stop the cart immediately until the obstacle is removed. However, it should be noted that the loads put on the cart must be secured well by the user, to avoid the risk of things falling down the edges when the robot stops suddenly due to loss of sight or due to the user stopping.

5.2.4 Financial Aspects

The following table shows a detailed cost analysis for each component in the project:

Table 11: Project Cost Analysts

Component name	No. of pieces/meters	Price per each (SR)
340KHz Mini DC-DC Step-down Converter	1	4
Raspberry pi 4 RAM 4GB	1	280
Power Bank 24000mAh	1	180
OAK-D camera	1	747
Electrical Hoverboard	1	150
IKEA-table	1	545
3d printing material	1	130
USB-C cable	1	80
SD-card 128GB	1	74.5
ESP-8266 node MCU	1	35
High visibility safety vest	1	45
LIPO Fireproof Safe Bag	1	120
Total costs (SR)		2,390.5

In general, we foresee this product will be useful and adopted by the following sectors/institutions:

- Airport operators
- Hotels
- Warehouses operators

The effects of our product locally are the increases in productivity and the positive impact on workers' long-term health, saving healthcare expenses. Globally, most of the first-world economies are facing or are going to face a labor shortage due to

demographic reasons. This product helps negate some of the productivity lost by allowing a smaller crew of workers to do more by appropriating partial automation of their tasks.^[16]

5.2.5 Social Impacts

We expect that our product will have to areas of impact on society. The two areas are habits, and beliefs. Starting with habits, a common habit that would change is the method of transporting items in indoor environments. Instead of carrying, pushing, or pulling heavy items, this task will become easier, safer, and more efficient to do. In the second area, the project will improve the view and acceptance on semi-autonomous robots for the better. This will pave the way for more sophisticated robots to be developed and adopted. However, this product can have an adverse effect if users didn't use it for its intended purpose, resulting in a reduction of the amount of exercise their bodies normally experience. In conclusion, the AutoCart product will have a positive impact directly through improving the long-term health of users and indirectly in changing their beliefs and views on robots from being a novel product to a practical everyday product.

5.3 CONCLUSIONS

In this project, the customer needs were to provide a method of transporting heavy objects indoors without the need to carry, push, or pull the things between the two points. The team provided a robot capable of carrying more than 50kg at once. The provided product can be controlled remotely, or can be used to follow the user, and even to walk in front of the user while the user control it with the an Aruco marker in the hand. To enable this product to move, the team used an electric hoverboard as the movement base and control it via serial communication.

The product was tested on different aspects, and it shows good results. The detection of the distance between the user and the robot had less than 1 cm error, and the communication was tested to show the reliability of ESP chip used for wireless communication. The hoverboard was also tested and proved it can provide consistent speeds even if the loads varied or increased even at 70kg. Finally, the product as a whole was tested and proved that it can follow the user

successfully, with some limitations in the follow mode, which can be effectively overcome using the other two modes. The project proved its ability to transport heavy objects between two points in closed environments reliably.

The team put a considerable effort in making the project look aesthetically appealing for the end users. The mechanical structure was bought from a famous manufacturer, and it is made of black aluminum and wood, to mix well in office environments. The team also tried to make some 3D printed compartments to help this goal. However, the product needs more work to reach the final market in its final form. This includes improvements to the following algorithm to do better following the user in corners and turns. More advanced improvements would be to develop an algorithm that doesn't depend on Aruco marker, to make it more convenient for the users. On the aesthetics side, the project needs more professional cable management for convenience and safety of the product and the users.

REFERENCES

- [1] P. Coenen, V. Gouttebauge, A. S. van der Burght, J. H. van Dieën, M. H. Frings-Dresen, A. J. van der Beek, and A. Burdorf, "The effect of lifting during work on low back pain: A health impact assessment based on a meta-analysis," in *Occupational and Environmental Medicine*, August. 2014. [Online] Available: https://www.researchgate.net/publication/265294967_The_effect_of_lifting_during_work_on_low_back_pain_A_health_impact_assessment_based_on_a_meta-analysis
- [2] Technical Regulation for Machinery Safety – Part 2: Mobile Machinery and Heavy Duty Equipment, Saudi Standards Metrology and Quality Organization SASO, May, 2021. [Online] Available: https://www.saso.gov.sa/en/Laws-And-Regulations/Technical_regulations/Documents/TR-Machinery-Safety-Part2-Mobile-Machinery-and-Heavy-Duty-Equipment.pdf
- [3] W. Ye, Z. Li, C. Yang, J. Sun, C. Su and R. Lu, "Vision-Based Human Tracking Control of a Wheeled Inverted Pendulum Robot," in *IEEE Transactions on Cybernetics*, vol. 46, no. 11, pp. 2423-2434, Nov. 2016, doi: 10.1109/TCYB.2015.2478154.
- [4] M. S. Hassan, A. F. Khan, M. W. Khan, M. Uzair, and K. Khurshid, "A computationally low cost vision based tracking algorithm for human following robot," in 2016 2nd International Conference on Control, Automation and Robotics (ICCAR), Apr. 2016.
- [5] N. A. Rawashdeh, R. M. Haddad, O. A. Jadallah and A. E. To'ma, "A person-following robotic cart controlled via a smartphone application: design and evaluation," 2017 International Conference on Research and Education in Mechatronics (REM), 2017, pp. 1-5, doi: 10.1109/REM.2017.8075245.
- [6] B. X. Chen, R. Sahdev, and J. K. Tsotsos, "Person following robot using selected online ADA-boosting with Stereo Camera," 2017 14th Conference on Computer and Robot Vision (CRV), May 2017.
- [7] Seung-hyeon Lee, Jae-won Choi, Ban Chien Dang, and Jong-wook Kim, "Development of Human Following Method of Mobile Robot Using QR Code

- and 2D LiDAR Sensor,” Journal of the Korean Society of Embedded Engineering, Vol. 15, No. 1, May. 2020.
- [8] Raspberry Pi, “Raspberry Pi 4 Model B,” Raspberry Pi 4 Model B Datasheet, June. 2019. [Online]. Available: <https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf>. [Accessed: 22-May-2022].
- [9] Espressif Systems, “ESP8266EX,” ESP8266EX Datasheet, October. 2020. [Online]. Available: https://espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf. [Accessed: 22-May-2022].
- [10] “BROR Trolley,” Storage Shelves and Units, IKEA. [Online]. Available: <https://www.ikea.com/sa/en/p/bror-trolley-black-pine-plywood-60333850/> [Accessed: 22-May-2022]
- [11] Emanuel Feru, “hoverboard-firmware-hack-FOC,” [Online]. Available: <https://github.com/EFeru/hoverboard-firmware-hack-FOC>. [Accessed: 22-May-2022].
- [12] “Detection of ArUco Markers,” OpenCV Documentation. [Online]. Available: <https://www.epa.gov/facts-and-figures-about-materials-waste-and-recycling/containers-and-packaging-product-specific-data>. [Accessed: 22-May-2022].
- [13] “Used Lithium-Ion Batteries,” United States Environmental Protection Agency. [Online]. Available: <https://www.epa.gov/recycle/used-lithium-ion-batteries> [Accessed: 22-May-2022].
- [14] A. Giles, “Is aluminium environmentally friendly?,” [Online]. Available: <https://origin-global.com/advice-centre/is-aluminium-environmentally-friendly>. [Accessed: 22-May-2022].
- [15] “Preventing fire and/or explosion injury from small and wearable lithium Battery Powered Devices,” Safety and Health Information Bulletin, OSHA, 20-Jun-2019 .[Online] .Available: <https://www.osha.gov/sites/default/files/publications/shib011819.pdf>. [Accessed: 22-May-2022].
- [16] “Competence centre on foresight,” Demographic trends of workforce | Knowledge for policy. [Online]. Available:

https://knowledge4policy.ec.europa.eu/foresight/topic/changing-nature-work/demographic-trends-of-workforce_en. [Accessed: 29-Nov-2021].

APPENDIX – A: VALIDATION PROCEDURES

EXPERIMENT 1

Title: Validation of Measurements' Accuracy On the project of "Auto Follower-Cart".

Conducted by: Ahmed Patwa | Member #2

Introduction

The Auto Follower-Cart is a follower robot used in indoor object transportation. It has two operation modes, one of which is the following mode, that allows it to follow the user. A camera on the robot detects a special marker called "Aruco marker" which the user will hold, and so the system can extract information about user's position using the marker. This involves distance and angle. If these measurements are not accurate, then it could lead to accidents and crashes with the environment. This experiment aims to how much of error these measurements produce to make sure they are reliable.

Objectives

- Measure the error in Aruco distance detection
- Measure the error in Aruco angle detection

Variables

- Distance of the ArUco marker from the camera
- Angle of rotation of the ArUco marker around the z axis (see the below figure to understand the axis)

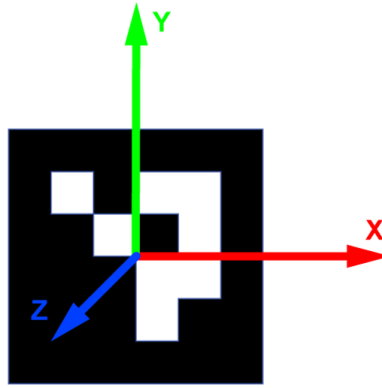


Figure 33: Axis of Aruco marker

Constants

- Lighting conditions.
- Position of the camera.

Assumptions

- It is assumed that the conductor of the experiment has calibrated his camera for Aruco pose-estimation prior to the experiment. Please refer to [1] for steps of calibration.
- It is assumed that the conductor of the experiment has an experience in working with Aruco markers and has an already has a running code for Aruco pose estimation. If not, please check [2] to get the code.

Safety

Since the experiment revolves around finding the reliability of Aruco distance and angle measurements, it is important that this experiment is conducted without connecting the code to the main project, because the main project is a moving robot, and it can't be used to control it until accuracy of measurements is validated.

Experiment tools

1. A computer.
2. A webcam.
3. A level tool.
4. A measure tape with a lock mechanism (at least 3-meters long).
5. A flat board.

6. Aruco marker with edge of 16 cm printed on A4 paper.
7. Duct tape

Obtaining distance work plan

1. Set the camera on a long, flat surface. Use the level tool to make sure it is actually flat.
2. Connect the camera to the computer and open the camera application to make sure it is running.
3. Use the leveling tool to make sure the camera is not tilted in any direction
4. NOTE: This step is very important, as tilting the camera may give unrealistic results.
5. Now, spread the measure tape in front of the camera, in a straight line, for 3 meters, and lock it so it doesn't fold back in.
6. Make sure the tip of the measure tape is parallel to the tip of camera lens.
7. Run the software on your computer.
8. Put the level tool on top of the measure tape, with its inner edge just after 50cm. Then, support the Aruco marker on the level tool to make sure it is leveled. Move them such that the Aruco marker is exactly 50 cm away from the camera. See the next graph for explanation of the setup.

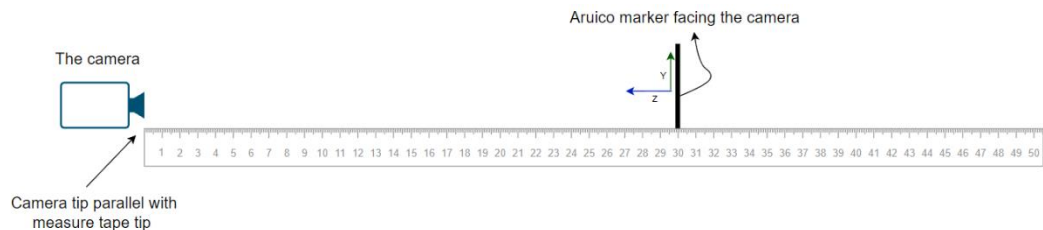


Figure 34: A graph to explain the setup for experiment 1 (distance)

9. Check the distance reading on screen (Labeled as z-axis). Record 10 readings of it.
10. Move the aruco marker 5 cm further then record 10 readings again. Repeat this until you reach 100cm. Then start going 10cm further up to 200cm. Then keep going 30cm up every time up to 290cm, and also record the distance at 300cm.

Obtaining angle work plane

1. Set the camera on a flat surface. Use the level tool to make sure it is actually flat.

2. Put the camera such that it is facing a flat surface. This could be a board or the wall.
3. Connect the camera to the computer and open the camera application to make sure it is running.
4. Use the leveling tool to make sure the camera is not tilted in any direction
5. NOTE: This step is very important, as tilting the camera may give unrealistic results.
6. Run the provided software. You will see the camera frame with 4 blue lines on it (like the following image). These will be used to align the Aruco with a certain angle.

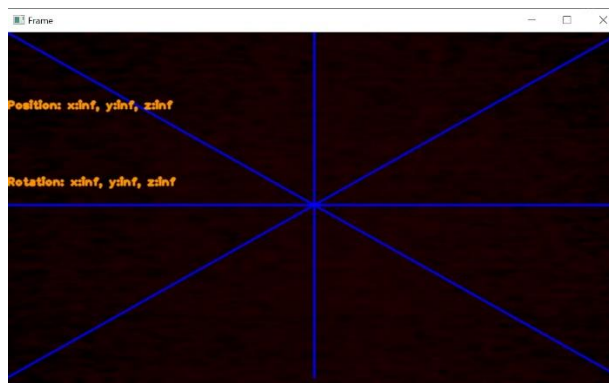


Figure 35: An example of the screen output (Experiment 1, angle)

7. Now, carefully align the Aruco so that it is completely straight, with no rotation. Use the lines on the screen to help you align it with the vertical lines in the Aruco marker. Then use the duct tape to tape it to the flat board surface.
8. Look at the screen and note the angle (Z-Axis angle). Record 10 readings.
9. Next, un-tape the Aruco, rotate it 45 degrees (use the lines on the screen to help you align it) then tape it on the flat surface again, and take 10 readings.
10. Repeat to obtain readings for angles: 0 (or 360), 45, 90, 135, 180, 225, 270, 315 degrees.

Collected data

Table 12: First Experiment, Distance Measurements - First batch

True Value (cm)	30	35	40	45	50	55	60	65
Measured (cm)	30.2475	35.4059	40.4342	44.7246	50.3576	55.6918	60.7555	65.2322
	30.1884	35.4464	39.9278	45.3799	50.3761	55.7293	60.7931	66.2137
	30.2197	35.4589	41.3105	46.3708	51.4262	55.7349	60.8007	66.2159

	30.2206	35.4062	40.4968	44.7840	51.3295	55.7216	60.7751	65.2849
	30.3436	35.5324	41.3545	45.3603	51.3545	55.7846	60.7958	66.2023
	30.3393	35.5633	41.4309	46.3606	49.8777	55.7163	60.9373	65.2309
	30.3679	35.6275	40.4034	45.3259	51.3593	55.8154	60.6058	66.2154
	30.3828	35.6285	39.8897	46.2327	49.9549	55.7964	60.7442	65.3657
	30.4218	35.6395	40.3679	46.2280	50.3243	55.7380	60.9450	65.2671
	30.2128	35.5162	40.3372	44.8723	50.3356	55.7832	60.8704	65.4309

Table 13: First Experiment, Distance Measurements - Second batch

True Value (cm)	70	75	80	85	90	95	100	120
Measured (cm)	69.9296	74.5820	80.7894	85.5244	89.5968	94.3426	101.0370	119.5549
	70.8257	74.7910	80.9720	85.4327	89.4621	94.4488	101.3974	119.2559
	69.8324	74.7718	80.7204	85.4281	89.3994	94.4419	101.2533	119.1025
	70.6736	76.6565	80.8767	85.2947	89.5843	94.3537	101.1110	119.5090
	70.4571	76.6342	80.7638	85.5085	89.3912	94.1610	100.8093	119.3309
	70.6913	75.8394	80.7453	85.2143	89.4347	94.1120	100.9362	119.0955
	70.3105	76.6651	80.5627	85.4610	89.6352	94.1070	100.8788	118.9897
	71.5856	75.7962	80.5660	85.3367	89.5097	94.0356	101.1286	118.6953
	71.6275	75.7739	80.5661	85.2321	89.5141	94.2479	100.9996	119.0647
	71.7192	75.6652	80.7114	85.2143	89.5331	94.0457	101.0002	118.8190

Table 14: First Experiment, Distance Measurements - Third batch

True Value (cm)	140	160	180	200	230	260	290	300
Measured (cm)	141.3156	160.3746	180.6731	201.0800	233.3486	265.6603	296.4744	310.0597
	141.5091	161.4347	180.4834	200.4581	233.0308	265.8470	295.7342	305.6110
	141.2269	160.9760	180.5828	200.5270	233.7146	264.2877	296.0681	303.5545
	140.7852	161.0868	180.6111	200.6122	233.0341	264.7643	297.0480	312.3788
	141.5546	160.7913	180.4083	199.9143	232.4752	264.5390	296.9467	305.7487
	141.0379	161.2268	180.6792	199.9270	233.4334	265.0853	295.7255	305.7551
	140.8232	161.0934	180.2749	199.9857	232.7415	265.4902	297.0450	311.6832
	141.1829	161.1778	180.9692	199.6739	232.9096	266.1924	297.0700	307.3218
	141.2861	161.8967	180.2560	200.6025	232.4039	264.8842	295.9283	307.0021
	141.4627	161.7194	180.3524	200.9712	232.9031	265.5428	295.7322	305.6006

Table 15: First Experiment, Angle measurements

True value (degrees)	0	45	90	135	180	225	270	315
Measured (degrees)	0.9428	46.2313	89.9252	136.0213	178.9242	225.5777	269.9091	315.8825
	0.8453	46.0467	89.7343	136.0213	178.9242	225.5777	269.9128	315.6448
	0.8660	46.0467	89.4384	135.9448	178.8780	225.6927	269.9279	315.5777
	0.6538	45.9773	89.7591	135.8077	178.9242	225.6369	269.9237	315.5777
	0.9005	45.9693	89.9228	135.8077	179.0539	225.4701	269.9237	315.6448
	0.6698	46.1647	89.8157	135.9390	178.8780	225.5582	269.9279	315.5777
	0.8660	46.0709	89.8157	136.0547	178.9321	225.5777	269.9279	315.6448
	0.7501	46.1169	89.9228	135.8077	178.9321	225.5777	269.9279	315.8825
	0.8660	45.9693	89.8157	135.7743	178.9600	225.6379	269.9279	315.8825

	0.7500	46.0005	89.8825	135.8077	178.9600	225.4701	269.8765	315.8825
--	--------	---------	---------	----------	----------	----------	----------	----------

Data analysis

The goal of the experiment is to measure how much error exists in the Aruco distance and angle measurements. Since we have multiple readings for multiple distances and angles, we will start by averaging each column in the data above, then calculating the error with formula:

$$\text{error}(\%) = \frac{|\text{True Value} - \text{Average measured value}|}{\text{True Value}} \times 100\%$$

The result is shown in the below tables:

Table 16: First experiment, Error calculations for distance measurements

True Value (cm)	Average measured distance (cm)	error(%)	error(cm)
30	30.2944	0.9815	0.2944
35	35.5225	1.4928	0.5225
40	40.5953	1.4882	0.5953
45	45.5639	1.2532	0.5639
50	50.6696	1.3392	0.6696
55	55.7511	1.3657	0.7511
60	60.8023	1.3372	0.8023
65	65.6659	1.0245	0.6659
70	70.7653	1.0932	0.7653
75	75.7175	0.9567	0.7175
80	80.7274	0.9092	0.7274
85	85.3647	0.4290	0.3647
90	89.5061	0.5488	0.4939
95	94.2296	0.8109	0.7704
100	101.0551	1.0551	1.0551
120	119.1417	0.7152	0.8583
140	141.2184	0.8703	1.2184
160	161.1778	0.7361	1.1778
180	180.5290	0.2939	0.5290
200	200.3752	0.1876	0.3752
230	232.9995	1.3041	2.9995
260	265.2293	2.0113	5.2293
290	296.3772	2.1990	6.3772
300	307.4715	2.4905	7.4715

Table 17: First experiment, Error calculations for angle measurements

True value (degrees)	Average Measured angle (degrees)	Error (%)	Error (Degree)
45	45.15935	0.354115	0.159352
90	89.80323	0.218631	0.196768

135	135.1986	0.147123	0.198615
180	180.1367	0.075917	0.136651
225	225.2762	0.122736	0.276157
270	270.1686	0.062428	0.168555
315	314.8087	0.060717	0.191258
360	359.7500	0.069454	0.250034

Discussion and conclusion

Looking at Table 12, Table 13, and Table 14, we can notice that the error is lower than 1cm for most of the data. This becomes more clear when we plot the amount of error, like shown in the next graph:

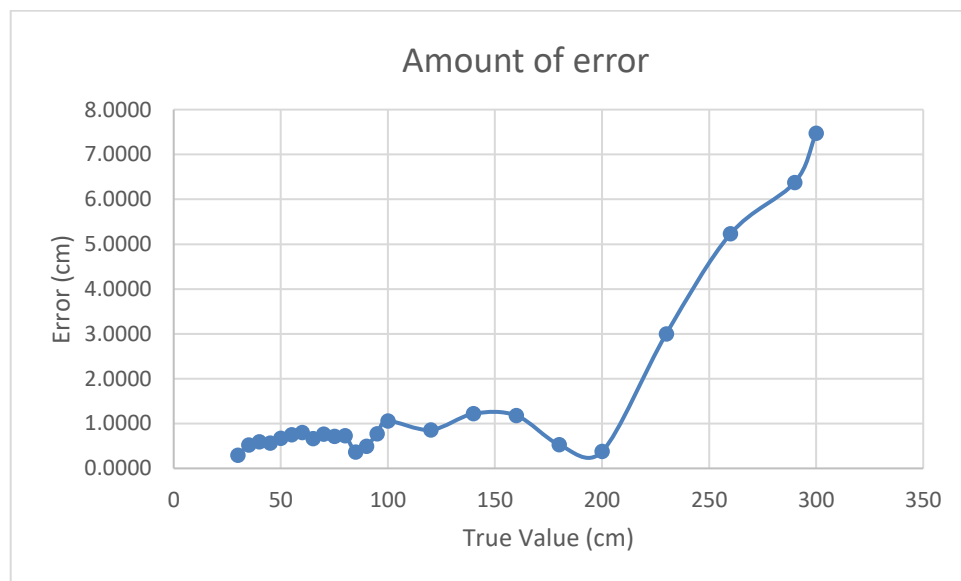


Figure 36: Plot of the error amount in distance measurements

From this graph, we notice that the error was less than 1cm at distances less than 100 cm. It also stayed less than 1.5cm for distances less than 200cm. However, it started to rise rapidly to reach +7cm error for 300cm distance. This means that the robot will see the user accurately and safely while he is closer than 2 meters.

On the other hand, the angle measurements appear to be more accurate according to the next graph:

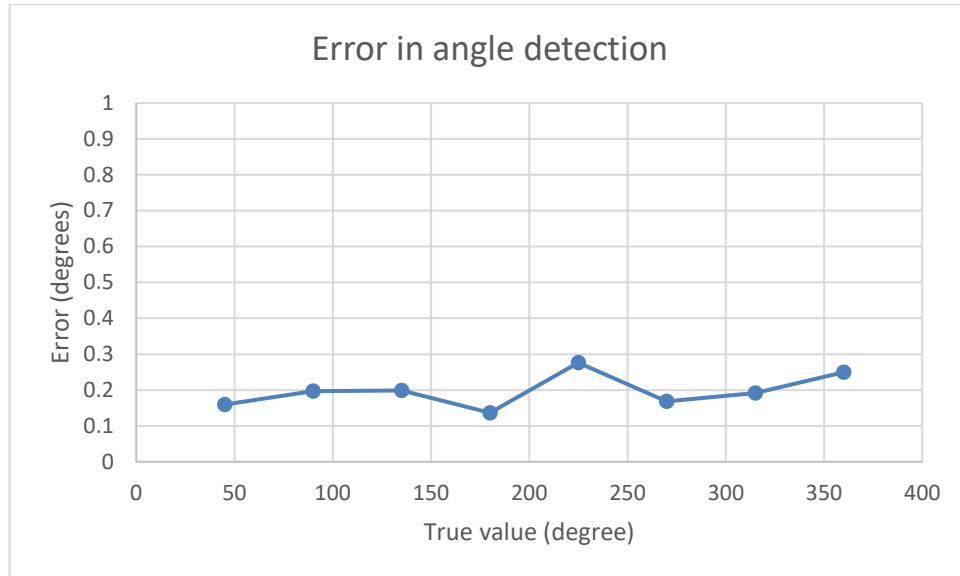


Figure 37: Plot of the error amount in angle measurements

We see some fluctuations indicating uncertainty. This might be due to poor alignment of the Aruco marker with the alignment lines in the setup. However, the error is still less than 0.3 degrees, which is considered safe for the purpose we are using angles in, which is steering the wheels in pushing mode, where the user will hold the Aruco marker in front of the camera and use it as a steering wheel.

Using the data collected in this experiment, we could conclude the reliability of using Aruco markers as a method for distance and angle detection to detect the position of the user and enable the robot to follow him.

Considerations of Engineering Standards

The results of this experiment enables us to be in accordance with SASO standards (Saudi Standards, Metrology, and Quality Organization). According to SASO, Pedestrian-controlled machinery control systems shall be designed in such manner that reduces the risks arising from unintended movement of the machine towards the driver. This is validated by checking the error margin of the method we used for this control. Other standards that we have been working on is the Signs, Signals and Warnings, which states that a moving vehicle shall have signs and warnings as needed to alert people of any problems arising. And even if the robot will not even be allowed to get close to the user while in following mode (as the results above has shown), this standard will be implemented as a warning system to warn the user whenever he is too close to the robot while in following mode, in case of any errors in detection.[3]

Experiment References

- [1] F. Souza, "3 ways to calibrate your camera using opencv and python," *Medium*, 29-Mar-2021.
[Online]. Available: <https://medium.com/vacatronics/3-ways-to-calibrate-your-camera-using-opencv-and-python-395528a51615> [Accessed: 21-Apr-2022].
- [2] Grab the code used in this experiment from the following link:
<https://github.com/aibtw/ArucoValidation>
- [3] Technical Regulation for Machinery Safety – Part 2: Mobile Machinery and Heavy-Duty Equipment Saudi Standards, Metrology and Quality Organization(SASO)-
https://saso.gov.sa/en/Laws-And-Regulations/Technical_regulations/Pages/default.aspx

EXPERIMENT 2

Title: Validation PID controller in Auto Follower-Cart under different loads.

Conducted by: Al Fahd Felemban | Member #3

Introduction

This is a validation report for the Auto Follower cart. The major part that is being validated is the ability of the PID control system inside the Auto follower cart to provide the same speed under different weight loads.

Tools

- Weight loads (used in exercises)
- Timer
- Meter
- Marker
- Our Auto follower cart



Figure 38 weights used in 2nd experiment



Figure 39 Second experiment, example of a load of 10 kg

Setup and work plan

Prepare a clear straight path of length [X]m (in this experiment we used 8m) and mark its start and end. Prepare the different weights then:

1. Place the cart at the start line.
2. Place weights on the cart starting with no load and then increase the weight with each iteration up to 70 KG.
3. Give a move command using the speed specified above 0.46 m/s.
4. Measure the time taken to reach the end line (take three reading for the same load).
5. Calculate and register the speed.

Data Collection

In our experiment, the weight load is varied, and the speed is measured. The measurements were repeated three times for each load for accuracy. Further, we will make statistical analyses of the collected data.

In this experiment, the speed command issued to the cart is to move at 0.46 m/s. It is assumed that the PID control will be able to maintain this speed under different weight loads.

Table 18: Experiment 2, data collection

Load (KG)	Speed m/s
0	0.45
0	0.454
0	0.445
10	0.445
10	0.454
10	0.441
16	0.447
16	0.446
16	0.442
20	0.438
20	0.441
20	0.441
25	0.443
25	0.439
25	0.446
30	0.446
30	0.441
30	0.442
45	0.448
45	0.446

45	0.441
50	0.445
50	0.441
50	0.443
75	0.441
75	0.447
75	0.44



Figure 40: Experiment 2, Average speed at different weights

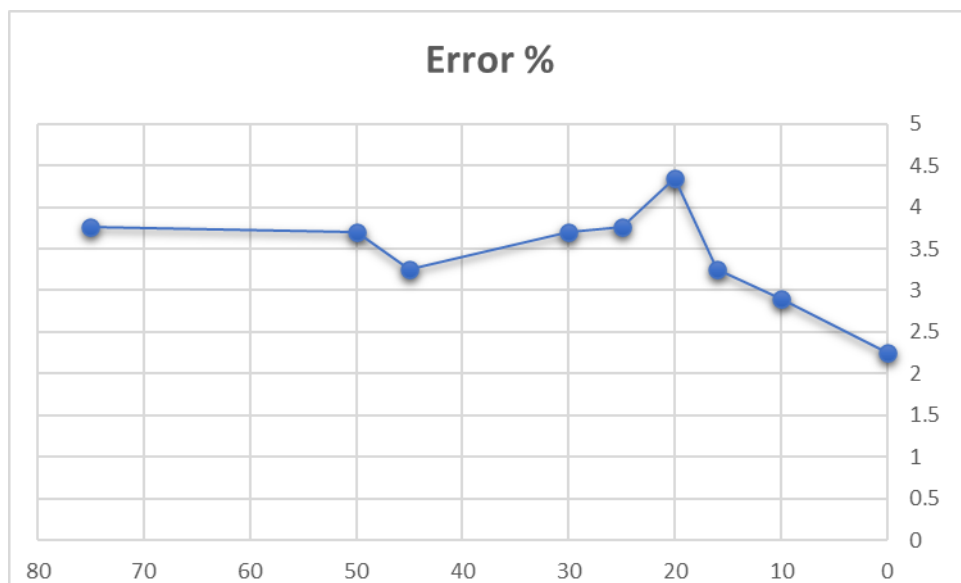


Figure 41: Experiment 2, Error for different weights. x-axis weights, and y-axis speed

Overall average speed under different loads is 0.444 m/s. the standard deviation is calculated to be 0.00263. As stated at the start of this experiment we want the cart to move at a speed of 0.46 m/s however, we can see that this speed was never reached. My assumption for why this happened is the following we used a

refurbished electrical hoverboard that was designed to have a user stand on it but in our implementation, the load is distributed over the table and not directly placed on top of it.

Conclusion

In this experiment the independent variable is the weight and the dependent variable is the speed. The results show a maximum error of 4.5%. in light of these results, we will add a 4.5% to the issued speed command to cover the error margin discovered in this experiment.

EXPERIMENT 3

Title: Validation of Wi-Fi Received Signal Strength

Conducted by: Amro Batwa | Member #1

Introduction

Our project, the Auto Follower Robot, is a robot that detects the user through a marker and as long as he is in line of sight, the robot will follow him in a close distance, it also features a remote-controlled mode, where the user can instead use to control the robot via his phone, or his portable computer (i.e. laptop or tablet), this feature is to solve the problem of following the user through a tight corridor or sharp angles, or a crowded room/facility with many obstacles in the Robot's way. In order to use the remote-control feature of our project the user's device has to establish a Wi-Fi connection with the Microcontroller used within the robot, which is ESP8266 – NodeMCU, which uses a Wi-Fi board based on IEEE 802.11b/g/n standard and capable of establishing 2.4 Ghz Wi-Fi connection.

This validation test will introduce measurement of the strength of this Wi-Fi signal, which will vary based on the distance, and the structure of user's environment. but before displaying our data, let us first understand the measurement of Wi-fi signal.

Wi-fi Signals are a type of Electromagnetic wavelengths which are transmitted through air as its medium. which are considered Radio frequency signals or RF. These signals are transmitted within frequencies 2.4 GHz and 5 GHz. To measure the received Wi-Fi signal there are different units that can be considered. Received signal strength indicator (RSSI), which used by manufacturers of Wi-Fi modules, however, this unit may not deliver the desired accuracy, this is because different modules use different values to measure their received signal. instead, we will be using Decibels in relation to the power (mW) sensed by the receiver. The following chart shows a general strength indication based on the dBm value:

Table 19: Third Experiment, Signal Strength Quality

Signal Strength in dBm	Assumed Quality
-30 dBm ~ -49 dBm	Perfect signal, indicates good quality and maximum transition speed of data
-50 dBm ~ -66 dBm	Very Acceptable Range, very high transition speed of data is expected

-67 dBm ~ -69 dBm	Still capable of establishing a connection that is sufficient for video streaming
-70 dBm ~ -79 dBm	Not strong signal, barely sufficient to maintain the connection, or send data in discrete form.
-80 dBm ~ -89 dBm	Insufficient signal, cannot be count on to transmit any form data.

for our case, Wi-Fi signal is important to ensure that control signals between the user-who may be positioned in various distances from the robot- are sufficient and accurate. failure in delivering accurate control signals might result in collision, which exposes the users or his property to physical damage. according to Saudi Standards, Metrology and Quality Organization, Supplier of mobile machinery is expected to eliminate or reduce potential risks as much as possible [3].

Objectives

To test and validate the values of signal strength in dBm relevant to distance, ensure the same results in different closed-environments, and using different devices. The units used for this experiment are:

- Signal strength measured in dBm
- Distance, measured in meter

Experimental Setup

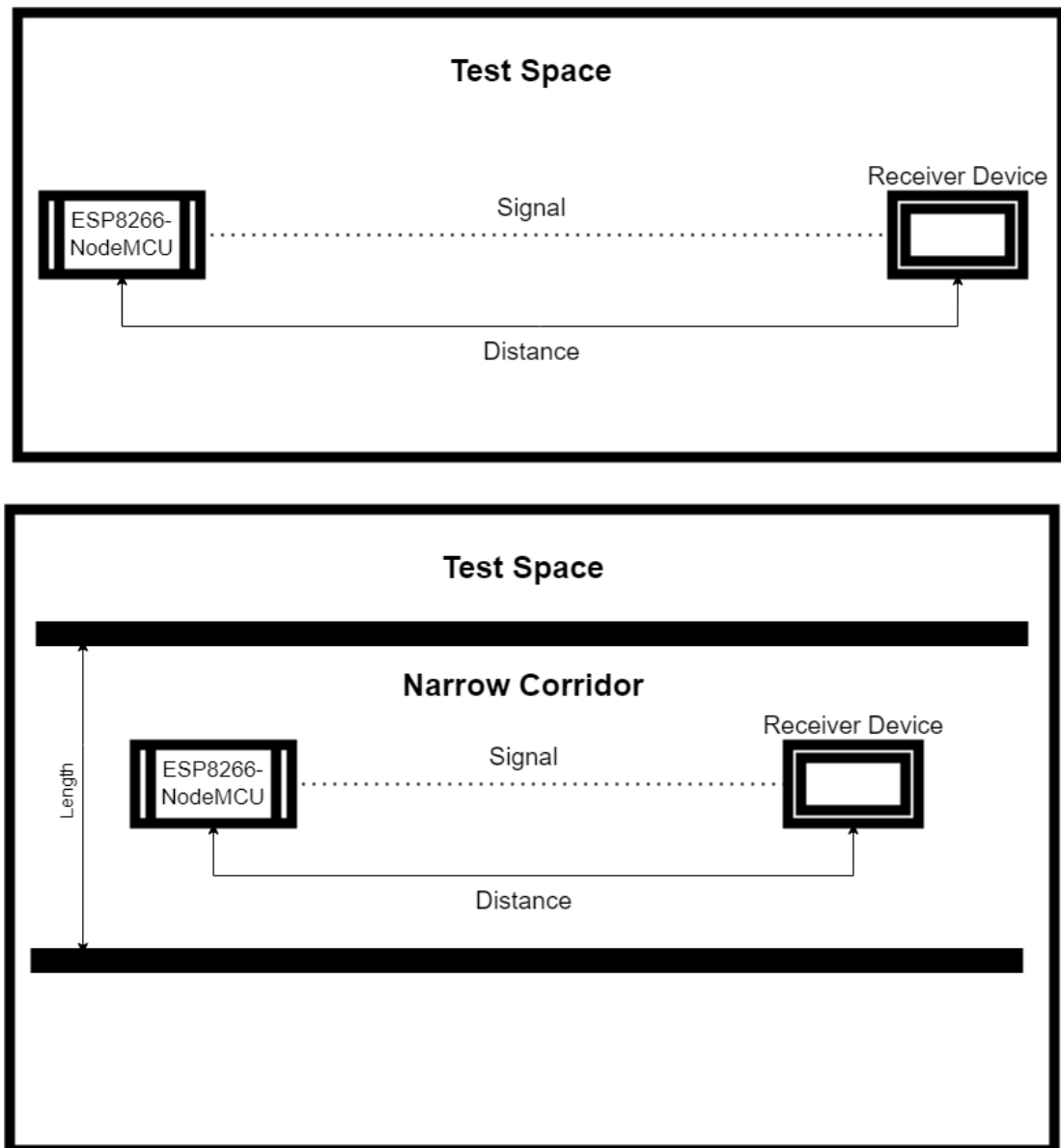


Figure 42: Third Experiment: Experiment Setup

Tools

1. ESP8266-NodeMCU
2. Distance measuring wheel
3. Laptop – Microsoft Surface Pro 7, using Wi-Fi 6: IEEE 802.11 a/b/g/n/ac/ax
4. Phone- Galaxy Note 9, using Wi-Fi 5 802.11 a/b/g/n/ac
5. NetSpot Application, which is an application provided on different platforms. NetSpot provides measurement of received signals in dBm as well as other statistical data.

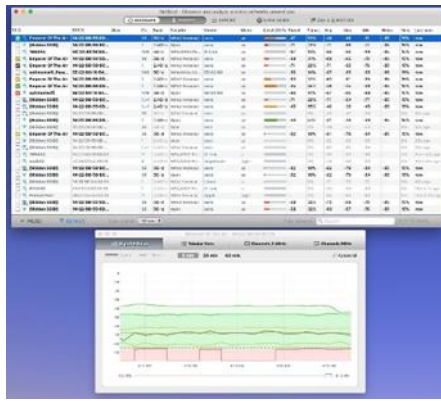


Figure 43: Third Experiment, NetSpot Application and measuring wheel

Work Plan

1. setup the ESP8266-NodeMCU as a server with a 2.4 GHz connection, which any device capable of establishing IEEE Wi-Fi connection should be able to connect on it.
2. install Netspot application.
3. use the measuring wheel to calculate distance from the position of ESP8266
4. position the receiver device at the measurement distance
5. run Netspot application and search for the ESP8266-NodeMCU Wi-fi signal (indicated by specified name or IP address)
6. remain for 20 seconds, then record the last 4 readings (readings interval is 5 seconds)
7. Record dBm signal showed by NetSpot
8. Repeat with different distances and different devices.
9. compare results and confirm difference margins.

Assumptions:

We assume that signal strength readings will be proportional to distance, where increasing distance will result in smaller values of dBm indicating weaker signal (interpretation of signal strength by dBm can be referenced at Table 19).

Issues and experimental hazards:

Since we are dealing with Wi-Fi signals, we are prone to signal interference, any forms of signal that could interfere with Wi-fi such as Microwave waves leakage, Thick physical obstacles, crowded signals of other Wi-fi or Bluetooth devices. in order to ensure the having optimal readings, such hazards should be avoided.

Collected Data

I conducted my experiment using the ESP8266-NodeMCU which is already installed in our robot. the experiment was done inside the King Abdul-Aziz CEIES Lab and its surrounding corridors. I received the following results:

Table 20: Third Experiment, Test: open space using laptop

Distance in meter	Approximate Average Signal Strength in dBm
1 m	-46 dBm
5 m	-54 dBm
10 m	-57 dBm
22 m	-70 dBm
30 m	-81 dBm

Table 21: Third Experiment, Test: Narrow corridors using phone

Distance in meter	Approximate Average Signal Strength in dBm
1 m	-48 dBm
5 m	-55 dBm
10 m	-57 dBm
22 m	-73 dBm
30 m	-82 dBm

we can no display both readings in a single scatter chart to view the difference between the two tests. the tabulated results can be fit in the following chart:

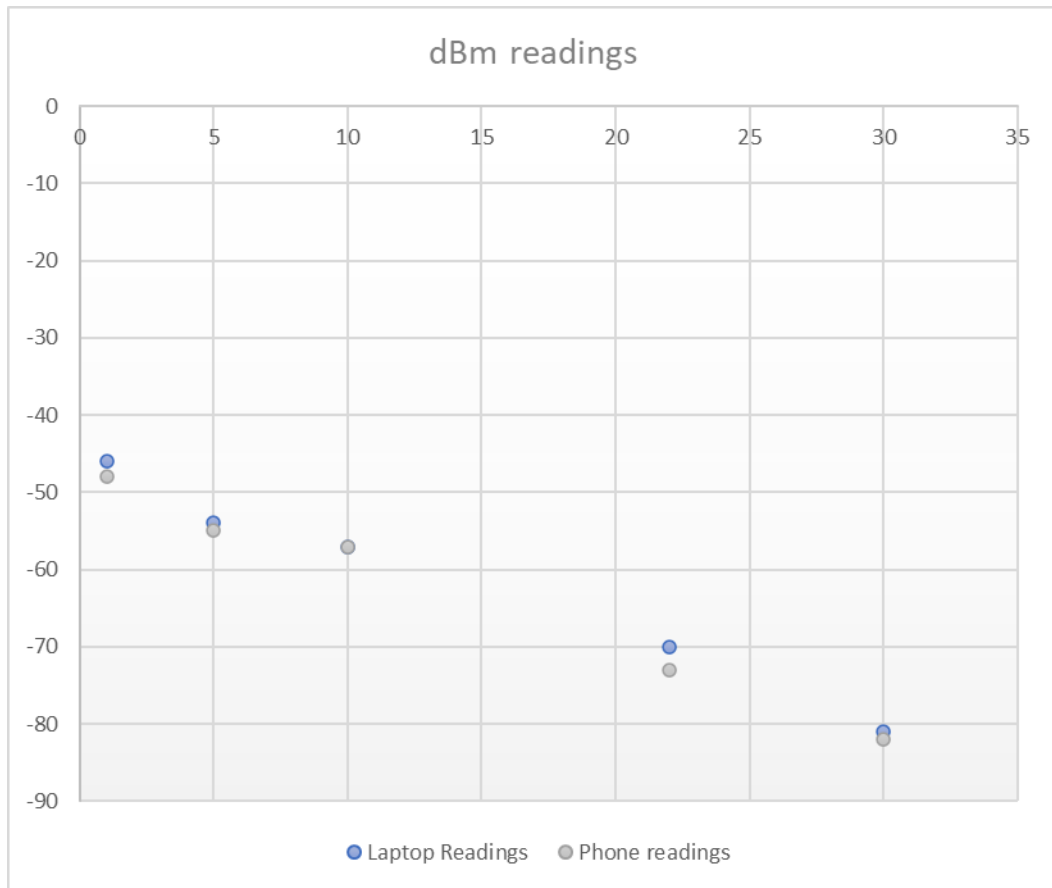


Figure 44: Third Experiment, dBm readings chart

If we are to apply our interpretation of signal strength from Table 19, we can view our readings in the following chart:

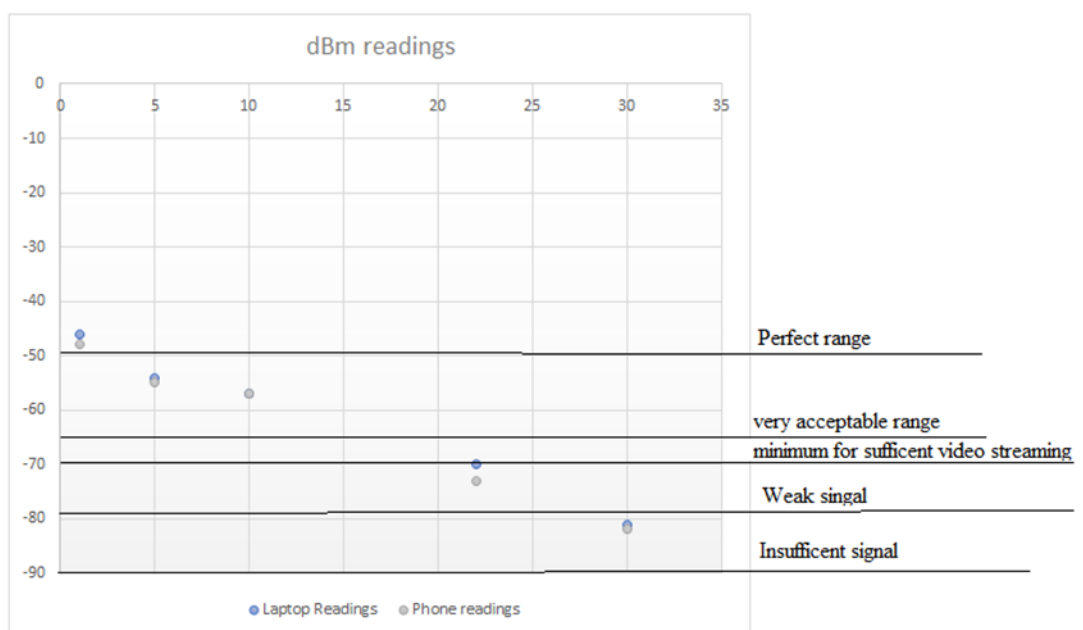


Figure 45: Third Experiment, dBm readings with signal strength levels

Conclusion:

from Figure 44, we can clearly see how the dBm values changed in accordance with the distance, the higher the distance, the weaker signal got. can also notice the difference between the phone and laptop readings. different devices have different networking modules, which results in a slight difference in signal strength. when going above 25 meters, the signal became very insufficient, no matter what device was used. this is a limitation based on Wi-fi technology, which we considered in our project design.

Based on this observation, and in accordance with the Engineering standards by The Saudi Standards, Metrology and Quality Organization, which states that the supplier of any controlled mobile machinery has to include clear instructions and detail the limitations of control system. our team will consider detailing all instructions that preserve and ensure the safety of users, and their propriety from any physical damage that can be caused by misusing the control system of our artifact. that includes the limitations of distance between the control device and robot as calculated in the above sections.

Experiment References

1. Differences in RSSI Readings Made by Different Wi-Fi Chipsets: A Limitation of WLAN Localization, Thomas Gallagher, Binghao Li, Andrew G. Dempster, and Chris Rizos - <https://ieeexplore.ieee.org/abstract/document/5955283>
2. Wi-Fi Signal Strength: What Is a Good Signal And How Do You Measure It, Jan Pedro Tumusok and Jorunn D. Newth - <https://eyenetworks.no/en/wifi-signal-strength/>
3. Technical Regulation for Machinery Safety – Part 2: Mobile Machinery and Heavy-Duty Equipment Saudi Standards, Metrology and Quality Organization(SASO)-https://saso.gov.sa/en/Laws-And-Regulations/Technical_regulations/Pages/default.aspx

EXPERIMENT 4

Introduction

After finalizing our project and porting to the Raspberry Pi 4, we conducted this experiment to test one of major tasks that our robot delivers, which is following the user and keeping up with the user speed. since we are using the hoverboard as the movement mechanical system, we are provided with high torque and responsive deacceleration, so in order to smooth out the movement to fit in closed environments, we implemented an algorithm that controls the speeds continuously which makes the robot moves smoothly with speeds proportional to the user's current speed. In this experiment our goal is test the efficiency of this algorithm.

In this experiment we will provide a practical test where we put weights of 20 kg on the cart and move several times for a fixed amount of distance. We will conduct the same test for distance of 10 meters and 20 meters. Each time the user will walk for the specified distance and compare the time he reached destination and the time the robot took to reach destination as well. Our main goal here is to test the consistency of the delay between the stopping timing of the user and the cart.

Objectives

The main objective is to test the capability of the robot to follow the user to his destination. We will measure the difference between the stopping time of the user and the cart after walking in normal speeds for a distance of 10 and 20 meters. the average walking speed tuned for closed environment per our testing was around 0.8 meters per second. the units we used for the experiment are:

- Fixed distances in meters
- Users stop timing in seconds
- Cart stop timing in seconds
- Stopping delay (User stop timing - Cart stop timing) in seconds

Tools

1. The AutoCart with the code running with following main parts:
 - a. Depth Ai camera.
 - b. nodeMCU ESP 8266.
 - c. Raspberry Pi.

- d. Hoverboard and the mechanical structure.
2. Measuring tape.
3. Ground markers.
4. Stopwatch.
5. Aruco marker vest

Setup and work plan

to setup for this experiment, we marked the ground for the required distances which are 10 and 20 meters. and setup the robot with following mode. The workplan is as follows:

- 1- Setup Follower Robot in Following mode.
- 2- Stand on the ground marker, with the robot behind you, and make sure the distance to the Robot is 1 meter.
- 3- Run the code, and start walking in average walking speeds (around 0.8 meters per second)
- 4- Record timing when user stops at the next ground marker (after 10 meters)
- 5- Record timing when the Follower Robot fully stops (it will stop 1 meter away from the user)
- 6- calculate the difference between user stop timing and Robot stop timing

Assumptions:

Our focus with algorithm that controls the speeding during following mode is to be as smooth as possible, meaning that the speed will not fluctuate suddenly in a manner that might spook the user or cause constant stopping and reaccelerating. we also assume that our stop timings will be not affected by the load carried or by acceleration/fraction delays that's due to the excellent build of self-balancing hoverboards which are designed to carry real people. We assume that the stopping delays are consistent in value in every attempt and by varying the distance moved, however due to variations of user's speed distribution during the walking, we assume a small fluctuation in stopping delay values.

Issues and experimental hazards:

One hazards to this experiment is that it is difficult to tune human walking to a certain speed, so instead we made this experiment by walking in normal speeds then measure its variations and average which was approximately 0.8 meters per second. it is also difficult to conduct this experiment alone, that is because if the user turned to see when the Follower Robot stops in order to time it, he might get the Aruco out of the Camera's frame which will cause it to stop earlier than intended and mess up the correct timing required for the experiment.

Collected Data

The following tables show our collected data by conducting the experiment as detailed in the workplan section.

Table 22: Fourth experiment, data collection (10 meters distance)

Attempt	User's stop timing	Cart's stop timing	Stopping delay
1	14.16	15.14	0.98
2	13.26	14.29	1.03
3	13.94	15.3	1.36
4	13.17	14.61	1.44
5	11.75	13.01	1.26
6	12.34	13.54	1.2
7	12.14	13.48	1.34
8	13.61	14.9	1.29
9	11.94	13.12	1.18
10	12.07	13.42	1.35

Table 23: Fourth experiment, data collection (20 meters distance)

attempt	User's stop timing	Cart stop timing	Stopping delay
1	24.07	25.12	1.05
2	23.86	25.05	1.19
3	26.48	27.6	1.12
4	27.68	28.91	1.23
5	25.85	27.24	1.39
6	26.14	27.2	1.06
7	27.47	28.51	1.04
8	26.44	27.6	1.16
9	25.12	26.5	1.38
10	25.53	27	1.47

In the following chart we plot the differences from the two tables together to visually interpret the consistency of stopping delays.

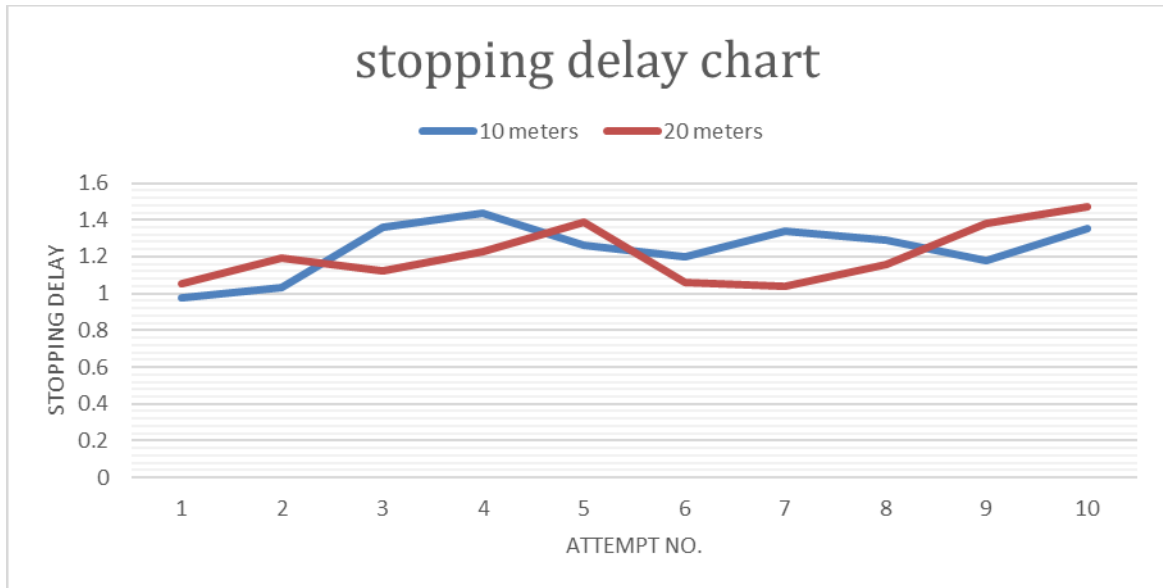


Figure 46: Fourth experiment, stopping delay chart

Conclusion:

from the chart and data tables we can see that the maximum delay was 1.47 seconds and lowest delay was 0.98s, and the standard deviation of the stopping delays is 0.148, meaning that the spread of the values of stopping delays has been not very high. this goes well with our expectations with the following algorithm where our aim was to deliver smooth movement with not much of unpredictable and sudden changes in speed .

In this experiment we also had the chance to use the finalized version of our project, giving us a chance to test the integrity of the systems and how well it performs in a practical situation. we can safely say that the results were satisfactory.

APPENDIX – B: SELF ASSESSMENT CHECKLIST

E: Exemplary, **S:** Satisfactory, **D:** Developing, and **U:** Unsatisfactory.

Student Outcome (SO)	Key Performance Index (KPI)	Self-assessment (E, S, D, or U)		
		M1	M2	M3
1. an ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics	1.1. Problem Identification	E	S	S
	1.2. Problem formulation	S	S	D
	1.3. Problem solving	S	S	S
2. an ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors	2.1. Design Problem Definition	S	S	D
	2.2. Design Strategy	D	S	S
	2.3. Conceptual Design	S	S	S
3. an ability to communicate effectively with a range of audiences	3.1. Effective Written Communication	S	E	S
	3.2. Effective Oral Communication	S	S	E
4. an ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts	4.1. Recognition of Ethical and Professional Responsibility	E	S	D
	4.2. Consideration of Impact of Engineering Solutions	S	S	S
5. an ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives	5.1. Effective Team Interactions	S	D	S
	5.2. Use of Project Management Techniques	D	D	S
6. an ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions	6.1. Developing Appropriate Experiment	S	S	S
	6.2. Conducting Appropriate Experiment	S	S	S
	6.3. Analysis and interpretation of Experiment Data and Drawing Conclusions	S	S	S
7. an ability to acquire and apply new knowledge as needed, using appropriate learning strategies	7.1. Effective Access of information	S	S	S
	7.2. Ability to learn and apply new knowledge independently	S	S	S